The Dissertation Committee for Feng Wang
certifies that this is the approved version of the following dissertation:

On Multihop Wireless Network Management:
Measurement, Modeling and Control

Committee:

Simon S. Lam, Supervisor

Lili Qiu, Supervisor

Mohamed G. Gouda

Aloysius K. Mok

Yongguang Zhang

On Multihop Wireless Network Management: Measurement, Modeling and Control

by

Feng Wang, B.S., M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2007

To my grandmother,

my parents,

and

my dear wife Xiaoli

Acknowledgments

I am thankful to many people during my doctoral study. They have helped me in many different ways. Of these people, my special thanks go to my co-supervisors, Prof. Simon S. Lam and Prof. Lili Qiu. I wouldn't have gone this far without their guidance and generous support.

I am grateful to the precious time Prof. Lam spent on me. His rigor, high standards, and extensive knowledge in computer networks have benefited me tremendously and will benefit my future career.

I owe huge gratitude to Prof. Qiu. Her enthusiasm, insightful thinking, and persistence on research quality have shed light for me on my work. She has given me enormous advice and taught me many skills during my research. She has also provided me equipments for conducting important experiments.

I am also grateful to Dr. Yongguang Zhang. I was inspired by him to start my research in wireless networks. I learned many basics from him, which have been very valuable for my later work.

I truly thank my committee members, Prof. Mohamed G. Gouda and Prof. Aloysius K. Mok, for their time, commitment and advice.

I'd also like to thank my colleagues and members of my research group, especially Yun Mao, Mi Kyung Han, Dong Young Lee, Yan Li, Yi Li, Eric Rozner, Yogita Ashok Mehta, and Jayesh Seshadri.

On Multihop Wireless Network Management: Measurement, Modeling and Control

Publication No. _____

Feng Wang, Ph.D.
The University of Texas at Austin, 2007

Supervisors: Simon S. Lam
Lili Qiu

Multihop wireless networks are becoming a new attractive communication paradigm owing to their low cost and ease of deployment. Managing multihop wireless networks, however, is especially challenging due to a fluctuating wireless medium, presence of wireless interference, and the increasing demand for them to scale to large sizes.

This dissertation tackles the multihop wireless network management challenges by systematically integrating measurement, modeling and control. On the measurement and modeling side, this dissertation develops a novel probabilistic region-based localization algorithm to accurately determine node locations with limited and noisy measurement information. The dissertation further develops a general model of wireless interference to estimate throughput and goodput between arbitrary pairs of nodes in the presence of interference from other nodes in a wireless network. Our model advances state of the art in interference modeling by (i)

estimating interference among an arbitrary number of senders, (ii) modeling unicast transmissions, and (iii) modeling the general case of heterogeneous nodes with different traffic demands. On the control side, we investigated one of the most important network control problems – design of routing protocols for large wireless networks. We developed a new routing protocol, Small State and Small Stretch (S4) to jointly minimize routing state and routing stretch. S4 uses a combination of beacon distance-vector based global routing state and scoped distance-vector based local routing state to achieve a worst-case stretch of 3 using $O(\sqrt{N})$ routing state per node in an N-node network. Its performance benefits are further demonstrated in extensive simulation and testbed experiments.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The past decade has seen enormous development in wireless technologies. The technology advances boost the growth of diverse wireless networks, from single-hop wireless networks (SWNs) to multi-hop wireless networks (MWNs). In SWNs, such as cellular networks and wireless local area networks (WLANs), every node is within one hop of a central entity (base stations, access points). Users only communicate with the central entity. SWNs require much infrastructure support, hence are expensive to deploy. In comparison, nodes in MWNs can communicate with each other over multiple hops. MWNs require no or little infrastructure support. They are easy to deploy and cost-effective. Examples of MWNs include mobile ad-hoc networks (MANETs), wireless sensor networks (WSNs), and wireless mesh networks (WMNs). MWNs provide a platform for a broad range of applications, both special-purpose (*e.g.* search and rescue, environment monitoring) and general-purpose (*e.g.* broadband wireless Internet access). Therefore they have attracted more and more interests from researchers, network designers, and users. However, MWNs are very difficult to manage, due to fluctuating wireless medium, the presence of wireless interference, and the increasing demand for them to scale to large sizes. This dissertation tackles the challenges in MWNs management by systematically integrating measurement, modeling and control.

1.1 Background on Multi-hop Wireless Networks (MWNs)

The initial research on MWNs started in the early 1970's when packet radio networks were studied. There were limited prototypes in labs and military departments. MWNs received much wider attention since late 90's, thanks to the IEEE standardization efforts and the commercial success in wireless networks. Currently, there are three prevailing types of MWNs: MANETs, WSNs, and WMNs. In this section, we give a brief background on these networks.

1.1.1 Mobile Ad Hoc Networks (MANETs)

A mobile ad-hoc network consists of a collection of "peer" mobile nodes that are capable of communicating with each other without help from a fixed infrastructure. Each node is an end user as well as a router. The interconnections between nodes may change on a continual and arbitrary basis. Nodes within each other's radio range communicate directly via wireless links, while those that are far apart use other nodes as relays in a multi-hop fashion.

MANETs are suited for scenarios where an infrastructure does not exist, *e.g.* in disaster recovery situations where existing communication networks are destroyed. It is much quicker to deploy MANETs than rebuilding the infrastructure in these scenarios. MANETs are also proper choices for communications on battle fields where military units may move constantly and multi-hop connectivity may be desired.

There has been extensive research on MANETs, especially the MAC, routing and transport issues [104].

1.1.2 Wireless Mesh Networks (WMNs)

Generally, the nodes in a WMN can be categorized into two classes [4]: *mesh routers* and *mesh clients*. Mesh routers are nodes at the core of a WMN. They are dedicated for relaying traffic. Some mesh routers are connected to Internet through wired links. These routers act as gateways between wireless users and Internet. Mesh routers may provide ethernet interfaces to users without wireless network interface cards (NICs). Mesh routers are usually static. Mesh clients are wireless nodes at user side. They are the sources and destinations of the data traffic. They also have the option to participate in routing. An example WMN is shown in figure 1.1.

WMNs can be deployed over a metropolitan area, or over a community neighborhood. WMNs target at general-purpose civilian applications, *e.g.* broadband wireless Internet access, community networking and intelligent transportation systems.

The unique application scenarios have driven much research [4] in WMNs, particularly on performance, scalability and reliability issues.

1.1.3 Wireless Sensor Networks (WSNs)

A wireless sensor network consists of potentially large number of sensors, which are small, low-cost, low-power, and resource-constrained devices. Same as in MANETs, operations of WSNs do not require infrastructure support. Sensors can propagate the sensed and partially-processed data over multiple hops. Furthermore, there are usually some sink nodes in WSNs, which are responsible for collecting

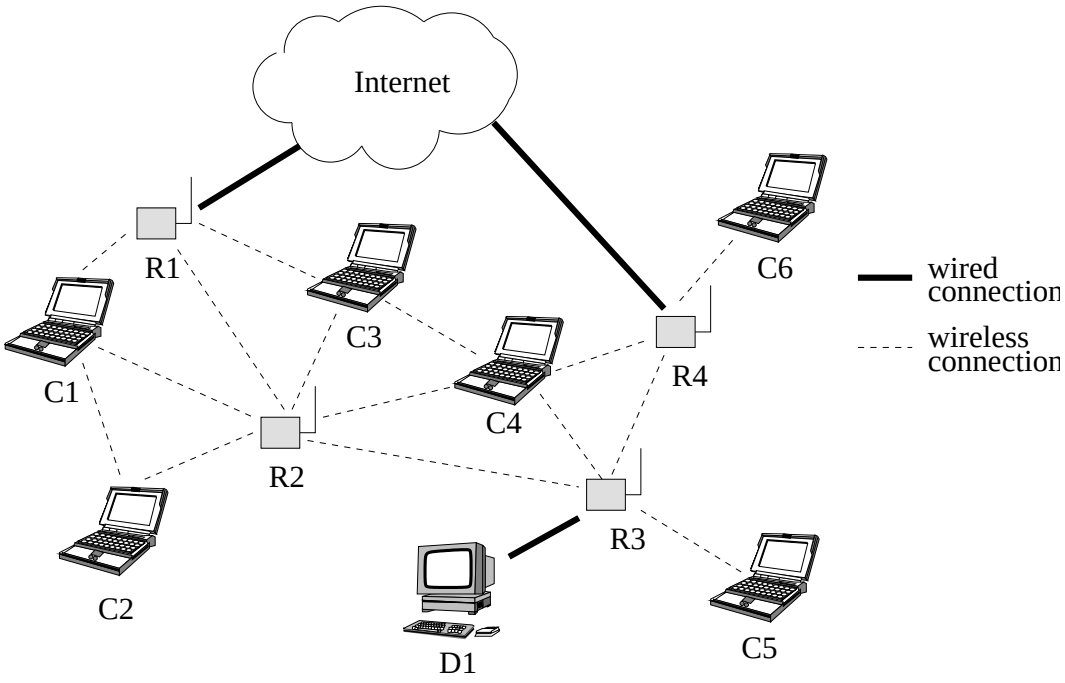


Figure 1.1: An example wireless mesh network. Three representative paths: (1) C2-R2-R1-Internet; (2) C1-R2-R3-C5; (3) C3-C4-R4-C6

the data. These sink nodes may send the data to a processing unit via other wired or wireless links.

WSNs are especially suited for environment monitoring in hazard or inaccessible places. Sensors are deployed densely and randomly in these places. WSNs can also be useful in health care to monitor and assist patients. Other applications include surveillance and targeting systems, smart home, etc.

Many research efforts have been made on WSNs. Especially, energy efficiency, fault tolerance and scalability [3] are among the active research topics due to resource constraints of sensors.

1.2 Motivation

Multihop wireless networks are becoming a new attractive communication paradigm owing to low cost and ease of deployment, and have found more and more applications. However, to fully achieve the promising features of MWNs, many research problems [3] [4] [11] still remain to be solved, such as network management, cross-layer protocol design and analysis, wireless security, etc. These problems may exist in other types of wireless networks too, but they become much more complicated in MWNs. In particular, *network management* is a challenging problem in MWNs due to multihop connections, wireless interference and increasing network scales.

1.2.1 What is Network Management?

Network management involves so many issues that it is hard to give a short definition. In [16], the author gives an intuition:

> "Intuitively, network management encompasses tasks associated with planning, deploying, configuring, operating, monitoring, tuning, repairing, and changing computer networks."

Network management, manual or automatic, is needed once a network is established. Simply put, there are three phases in managing a network: *measurement*, *modeling*, and *control*, as figure 1.2 shows. A management process may involve one or more of three phases.

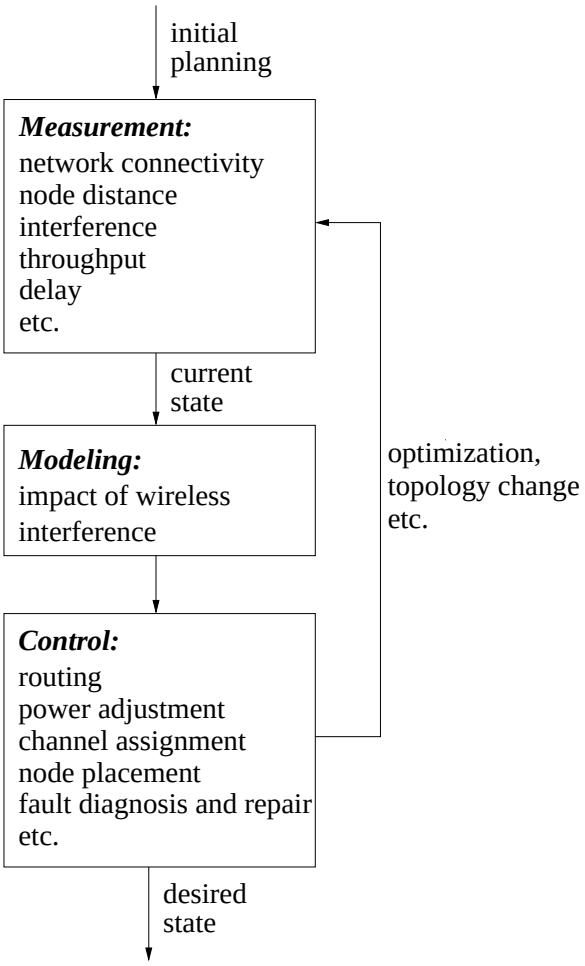


Figure 1.2: Three phases in network management

After the initial planning and deployment, network management may start with measuring the current network state, *e.g.* topology (network connectivity, node distance, interference, etc.), traffic (sources and destinations, transport protocol, etc.), and performance (throughput, delay, etc.). This phase collects information about how the network performs under current configurations.

The modeling phase abstracts and formalizes the general rules about the interrelationships, *e.g.* the impact of wireless interference, among the nodes and links. These rules may apply to any arbitrary network, not necessarily limited to the specific network of interest. Modeling phase takes measurements from real networks as input. With modeling, the managing entities can derive information that is otherwise expensive or difficult to measure directly, *e.g.* physical locations of nodes. Modeling also provides managing entities the capability to apply what-if analysis, and infer the performance under other configurations. Such analysis and inference can help managing entities determine the desired network configuration from possible choices.

Finally, the managing entities control the network behavior based on the output from the modeling phase, *e.g.* tuning or repairing the network to improve performance or achieve quality of service (QoS) requirements. Possible control parameters are routing mechanisms, channel assignment, transmission power, node placement, etc.

The above process of network management may repeat over multiple iterations for optimization purpose. It may also repeat because the network conditions change due to link failures, traffic turning on or off, environmental changes, etc.

1.2.2 Challenges in MWNs Management

The difficulty of network management increases dramatically along with the complexity of the network structure and traffic. Particularly, MWNs pose new challenges to all three phases in network management.

Multihop connections: Compared to nodes in SWNs, nodes in MWNs are connected in multiple hops. Moreover, MWNs do not have sufficient infrastructure support. Consequently, measurement tasks such as localization are more difficult to accomplish in MWNs than in SWNs, since most nodes are not within communication ranges of central entities.

Wireless interference: Understanding the impact of interference on real networks is extremely difficult, since interactions among nodes in MWNs are more complicated than in SWNs. Each node can be a source, destination, or even a router. Different nodes may have different traffic demands. Due to phenomenons such as hidden terminal and exposed terminal problems in MWNs, transmissions from one node may potentially affect not only neighboring nodes, but also nodes far away. Therefore it is much more difficult to model the impact of interference.

Large scale: MWNs such as WSNs and WMNs are expected to span a large scale, with respect to both number of nodes and size of the coverage area. WSNs may consist of tens of thousands of sensors, while WMNs may cover thousands of nodes in a metropolitan area. Controlling such a large-scale network, particularly routing, is a challenging issue.

1.3 Contributions of the Dissertation

This dissertation tackles the challenges in MWNs management by systematically integrating measurement, modeling and control. On the measurement and modeling side, this dissertation addresses *localization under multihop connections* and *modeling of wireless interference*. They do not directly change the way MWNs operate, but provide useful information for control of networks. On control side, this dissertation addresses *routing in large scale MWNs*.

The contributions of this dissertation are a set of approaches to MWNs management. Specifically, we propose (1) probabilistic region-based algorithms for localization under multihop connections, (2) a general model that captures impact of wireless interference and predicts network throughput and goodput, and (3) a small state and small stretch routing protocol that achieves scalability, efficiency, and reliability.

1.3.1 Probabilistic Region-Based Localization

Localization is to determine the physical locations of wireless nodes in the network. Providing location service is desired in MWNs management. It can improve performance of MAC and routing protocols. It also enables location-dependent applications. Centralized localization approaches exist to obtain location information. But they either are too costly (*e.g.* GPS [34]), or target at single-hop localization (*e.g.* activeBadge [105], RADAR [6], VORBA [73]). In this dissertation, we focus on distributed localization in MWNs. Although many distributed localization algorithms have been proposed [42, 95, 96], the following three top-

ics require further study: First, developing accurate localization algorithms using only connectivity information. Node connectivity is usually correlated with node locations. However, connectivity constraints may not be sufficiently strong to pinpoint a node. Therefore assigning the location of a wireless node to a single point may result in significant error. Second, leveraging additional information on location constraints. Localization accuracy relies heavily on the amount of available information about location constraints. To further improve accuracy, it is important to identify and exploit additional information on location constraints. Third, enhancing robustness of localization against erroneous information. Robustness is essential to the success of any localization scheme since erroneous measurement reports may arise from measurement errors, loss of measurement data, and hardware/software problems.

Our solution to localization problem has following three novelties. First, we develop probabilistic region-based localization algorithms, including using static grids, dynamic meshes, and segments of grids. Second, we propose several techniques to extract and leverage additional information on location constraints. The additional information can be applied to both our and others' localization schemes. Third, we develop techniques to enhance robustness of localization.

In our probabilistic region-based approach, we use a region to represent a node's estimated location. Each node derives a probability distribution over a set of cells in its region that it can possibly reside in. Every cell is associated with a probability about the likelihood that it contains the true position of the node. Starting from a small set of anchor nodes whose locations are known (*e.g.* from

GPS), our approach iteratively refines the probability distributions using location constraints, such as connectivity measured from the underlying network. A location constraint from node A to node B gives a probability distribution over the cells of B. The final probability of each cell is the product of the probabilities derived from all constraints, followed by normalization.

We also propose to measure and leverage additional location constraints: (i) network connectivity under different transmission power levels, which gives finer distance constraints, (ii) knowledge of carrier-sensibility [2], which gives extra distance constraint, (iii) layout maps, which restrains possible regions, and (iv) connectivity under directional antennas, which gives angle constraints.

Furthermore, our probabilistic region-based localization can be extended naturally to handle measurement errors and enhance robustness: the probability computation can take into account of the extent to which the location constraints are satisfied. In this way, a mesh cell that is inconsistent with most location constraints is assigned a low probability and pruned out, whereas a mesh cell satisfying most location constraints (but not necessarily all the constraints) will still be retained.

We evaluate our localization approach with extensive simulations. The results show that our approach provides a wide range of trade-off between accuracy and computation costs, making it suitable for difference types of MWNs, such as WSNs and WMNs. The results also verify additional location constraints can significantly improve the accuracy. Moreover, we demonstrate our enhanced scheme for robustness can achieve high accuracy even in the presence of significant measurement errors.

1.3.2 A General Model of Wireless Interference

Wireless interference is fundamental to the performance of wireless networks. Understanding wireless interference is essential to the design and management of wireless networks. However, it is extremely difficult to accurately estimate interference and its impact on network performance due to the complicated interactions among nodes in real networks. Despite significant progress on modeling wireless network performance, several important issues need to be addressed in order to accurately model wireless interference. First, the existing models for general network topologies can only handle two broadcast senders or two flows. Modeling wireless interference in the presence of an arbitrary number of senders is significantly more challenging due to the complex interactions among different nodes. Second, the existing models for general network topologies only consider broadcast traffic. A unicast transmission is more common, but it involves transmissions in two directions: data and ACK. Hence modeling unicast transmissions introduces additional complexities. Third, real networks often consist of heterogeneous nodes with different traffic demands and different radio characteristics. It is therefore essential for the interference model to support such heterogeneity.

To study wireless interference and its impact, we develop a general interference model that allows us to accurately estimate the throughput and goodput from real measurements in static multi-hop wireless networks. Compared to existing measurement-based models, our model advances the state of art in three important ways. First, it goes beyond pairwise interference and estimates interference among an arbitrary number of senders. Second, it goes beyond broadcast transmissions and

models the more common case of unicast transmissions. Third, it can accurately model interference in a heterogeneous environment with arbitrary (non-saturated) traffic demands, asymmetric link quality, and non-binary interference relationships.

Our model captures the interdependencies among the transmissions and receptions of all nodes. The inputs to the model are: i) traffic demand from each sender to each receiver , and ii) RF profile, which refers to the received signal strength (RSS) between every pair of nodes. The outputs are the throughput and goodput, normalized by MAC layer data rate. We obtain the RF profile from simple measurements. For an N-node network, our model requires $O(N)$ measurements, the minimum to build an RF profile for all N nodes. In the measurement phase, each node broadcasts in turn, while other $N-1$ nodes listen and record received signal strength index (RSSI) information for each received packet. From these measurements, we recover pairwise RSS and background interference due to sources other than nodes in the modeled network (Section 3.7). Then, we apply our *sender model* to estimate the amount of traffic sent by each sender under the given demand and our *receiver model* to estimate the amount of traffic successfully received. For saturated broadcast demands, our model can estimate throughput and goodput by computing the stationary probabilities of a Markov model. For unicast demands or unsaturated broadcast demands, the transition matrix of the Markov model involves additional variables and its stationary probabilities are solved in an iterative framework.

To validate our model, we conduct extensive simulations using the Qualnet simulator [84] and real experimental measurements from wireless testbeds. The results show that our model gives accurate prediction over a wide range of scenarios.

1.3.3 Small State and Small Stretch Routing

Routing is a functionality of finding paths and controlling the packets to follow them. The effectiveness of routing protocols directly affects network scalability, efficiency, and reliability. With continuing growth of MWNs scales, it is difficult yet important to develop routing protocols that *simultaneously* achieve the following design goals: First, small routing state. Using small amounts of routing state is essential to achieving network scalability. It reduces the storage requirement to form large networks. It also helps to reduce control traffic in route setup and maintenance, since the amount of routing state and control traffic are often correlated. Second, small routing stretch. Routing stretch is defined as the ratio between the cost of selected route and the cost of optimal route. Small routing stretch means that the selected route is efficient compared to the optimal route. Given the limitation on routing state, it is a challenging issue to achieve small routing stretch. Third, resilience. Wireless networks often experience frequent topology changes arising from link failures, and environmental changes. How to find efficient routes for instant recovery is difficult.

We present a new routing protocol, Small State and Small Stretch (S4), which jointly minimizes the state and stretch. S4 is a unique addition to the routing protocol design space. It is the first routing protocol that achieves a worst-case constant routing stretch of 3, using $O(\sqrt{N})$ routing state per node, in an N-node large scale wireless networks. It significantly enhances network resilience employing a distance guided local failure recovery scheme.

S4 exploits the theoretical ideas of the compact routing algorithm [101]. In

S4, a subset of nodes are chosen as beacon nodes. Each node maintains a local cluster according to its distance to the closest beacon. S4 consists of the following three major components: (i) scoped distance vector for building and maintaining routing state to nodes within a cluster, (ii) resilient beacon distance vector for efficient routing towards beacon nodes and facilitating inter-cluster routing, and (iii) distance guided local failure recovery for providing high quality routes even under dynamic topology changes.

S4 starts with measurement phase, in which nodes learn about network topology. First, beacon nodes broadcast global beacon messages to the whole network. Each node measures its "distance" to a beacon upon receiving a beacon message. Then, each node sends scoped cluster messages, with the scope being its "distance" to the closest beacon. Upon receiving any cluster message, a node adds the source of the message into its local cluster.

With the above measurements of topology, each node can control the routing to achieve scalability, efficiency, and reliability. Each node maintains state, *i.e.* next hop and cost, of optimal routes to beacon nodes and nodes in its local cluster. The routing criteria are as follows: if a destination is within the local cluster (intra-cluster routing), a node forwards packets along the optimal route; if a destination is outside the local cluster (inter-cluster routing), a node first forwards packets along the optimal route to the beacon closest to the destination, and then to the final destination. When a route failure occurs, the forwarding node broadcasts a recovery request message to neighbors. Upon receiving recovery reply messages, the forwarding node chooses the neighbor closest to the destination for recovery.

We evaluate S4 using both simulations in TOSSIM, a packet-level simulator for large-scale WSNs, and experiments in a 42-node WSN testbed. TOSSIM simulations fully stress the scalability, while the testbed experiments evaluate S4 under realistic RF and failure dynamics. The results show that S4 is scalable, efficient, and resilient to failures in a wide range of scenarios.

1.4 Organization

This dissertation is organized as follows. Chapter 2 presents the probilistic region-based localization approach. We start with basic algorithm and then propose techniques to improve accuracy and computation cost, extensions to leverage additional location constraints, and enhancement to achieve robustness. We show evaluation results from simulations. Chapter 3 introduces the interference modeling problem and presents a general measurement-based model. Experiment results are shown to verify the accuracy of the model. Chapter 4 presents S4 routing protocol. We first describe theoretical idea and the three major components of S4. Then we evaluate it using both TOSSIM simulations and testbed experiments. Finally, Chapter 5 concludes and gives possible directions for future research.

Chapter 2

Probabilistic Region-Based Localization

2.1 Overview

Determining the physical location of wireless nodes is important to a wide variety of applications, ranging from geographic routing [45, 86] to context-aware applications [52, 55], from habitat monitoring [13] to environment surveillance [5, 98].

A global positioning system (GPS) [34] can be used to obtain location information. But it requires line-of-sight communication with the satellites, and does not work in either indoor or outdoor environments with lots of obstacles (*e.g.*urban area). It is also costly to equip every wireless node with GPS. The limitation of GPS has motivated researchers to develop algorithms to infer location using cheap hardware by leveraging network connectivity, signal strength, and angle-of-arrival information [6, 40, 42, 68, 73, 95, 96, 105]. Despite extensive research in the area of localization, the following three topics in localization research require further study.

First, developing accurate localization algorithms based on only connectivity information is an active research topic. A major factor that determines the effectiveness of the algorithms is how the estimated locations are represented. In many previous studies, the location of a node is estimated as a single point. As

shown in [24], there are often many coordinate assignments that satisfy the location constraints derived from an underlying network. Therefore assigning the location of a wireless node to a single point may result in significant error. For example, as described in [35], when a node is constrained to be located at four corners of a region, a single point estimation may place the node at the center, which is misleading. In addition, a single point representation is vulnerable to measurement errors – a small perturbation in measurement data may result in a large difference in the estimated location [71]. The novel approaches, proposed by Galstyan et al. and Guha et al. [29, 35], are to represent the estimated location as a region that consists of all points satisfying the location constraints. Such a region-based representation has the potential to yield higher accuracy.

Motivated by [29, 35], we also use a region to represent a node's estimated location. To achieve even higher accuracy, we propose a probabilistic localization approach. In this approach, each node derives a probability distribution over a set of cells that it can possibly reside in. Every cell is associated with a probability about the likelihood that it contains the true position of the node. Furthermore, we propose two techniques to reduce computation cost. The first technique combines cells into segments, which significantly reduces computation cost with a moderate increase in localization error. The second technique is called probabilistic dynamic mesh-based localization (PDM). It uses a mesh generator to partition a region into a mesh, and represents the estimated location of a wireless node as a set of mesh cells. It iteratively refines the estimated location using location constraints extracted from the underlying network. It achieves high accuracy by deriving the probability

distribution of a node's position over the region. It achieves reasonable cost by adaptively changing the mesh cell size using DistMesh [21], which is an efficient way to generate an unstructured triangular and tetrahedral mesh to cover a region.

Second, localization accuracy relies heavily on the amount of available information about location constraints. For example, as shown in [23], there is a fundamental limit in localization accuracy using commodity 802.11 hardware. To further improve accuracy, additional information on location constraints is necessary. In this dissertation, we propose the following ways to obtain and leverage additional information: (i) using network connectivity under different transmission power levels, (ii) using knowledge of whether two nodes can sense each other's carrier, which can be measured empirically as shown in [2], (iii) using layout maps, and (iv) using more powerful anchor nodes (*e.g.*, the anchor nodes can not only extract distance constraints for its neighbors, but also obtain the approximate angles). We also evaluate the benefit of each type of such additional information.

Third, the *robustness* issue in localization has received little attention, even though robustness is essential to the success of any localization scheme since we cannot expect that measurements are always accurate. Erroneous measurement reports may arise from measurement errors, loss of measurement data, and hardware/software problems. Our probabilistic region-based localization provides a natural mechanism to handle measurement errors – the probability computation can take into account of the extent to which the location constraints are satisfied. In this way, a mesh cell that is inconsistent with most location constraints is assigned a low probability and pruned out, whereas a mesh cell satisfying most location constraints

(but not necessarily all the constraints) will still be retained.

In summary, while localization has been an extensively studied subject, our approach has the following three novel contributions. First, we develop probabilistic region-based localization algorithms, including using static grids, dynamic meshes, and segments of grids. These algorithms provide a wide range of trade-off between accuracy and cost. For example, the segments-based approach yields low cost and high accuracy, and is well suited for networks formed by less powerful nodes, such as sensor networks. In comparison, the PDM achieves a higher accuracy at a higher cost, making it suitable for networks formed by more powerful nodes, such as mesh networks. Second, we propose several techniques to extract and leverage additional information on location constraints. The additional information can be applied to both our and others' localization schemes. Our results show that the additional information can significantly improve localization accuracy. Third, we develop techniques to enhance robustness of localization, and show that the enhanced algorithm can tolerate significant errors from measurement data.

2.2 Related Work

Localization has been extensively studied due to its great importance. We broadly classify previous work into the following four areas: (i) localization over single-hop wireless connections, (ii) localization over multihop wireless connections, (iii) analysis of the fundamental limitations of localization schemes, and (iv) controlling node placement to ease localization.

Localization over single-hop wireless connections: For localization over single-hop wireless connections, nodes are one hop away from anchor nodes whose locations are known. Many approaches have been proposed in this area. Most of them localize users or hosts inside an office building. The general procedure is similar for all approaches: first, estimate relative locations (*e.g.*distance, angle) of nodes with respect to anchors; then, infer absolute locations from relative locations. However, there are many different ways to estimate distance or angle: ranging with infrared, radio signal, ultrasound; Time Difference of Arrival; Angle of Arrival, etc. Accordingly, the techniques applied to infer locations can be different as well, such as trilateration, triangulation, data-fitting, etc.

Active Badge [105] is a location system for in-building localization of office staff. Staff members wear infrared (IR) badges that periodically transmit unique identifiers. Sensors are deployed as anchors around the building to receive IR signals from badges. A central location server polls sensors about signals from badges and determines locations of staff members based on which sensors hear the signals from their badges. Active Badge locates staff members to the level of rooms where they are sighted. It requires support from extra infrastructure of sensors, which may incur high cost of deployment.

RADAR [6] tackles the problem of localization for radio frequency (RF) based wireless networks in in-building environments. It relies on signal strength measurement gathered at multiple receiver locations to determine locations of users by triangulation. Three base stations record signal strength of beacons from hosts. In off-line phase, signal strength information as a function of location is collected.

Then in real-time phase, the base stations measure signal strength from a host to be localized, and infer its location that best matches the collected data. The accuracy of RADAR depends on the accuracy of the signal strength map collected in off-line phase. When the signal strength varies significantly, *e.g.*due to people moving around, it may require to measure a new signal strength map.

Cricket [83] is another location system for in-building applications. Beacons are deployed at strategic locations inside a building. Beacons send out messages to advertise their location information. Each mobile or static device is equipped with a listener. Listeners receive messages from beacons and infer their own locations from these messages. For a listener to determine the distance to beacons, Cricket uses a combination of RF and ultrasound signals. Each time a beacon advertises its location over an RF signal, it concurrently sends an ultrasonic pulse. A listener listens to both RF signals and ultrasonic pulses, and uses the difference between the arrival time of radio and ultrasound signals to estimate distance to beacons and further infer its location. As in RADAR, the variations of RF signal strength may also affect the accuracy of Cricket. In addition, the placement of beacons is nontrivial and increases cost.

VORBA [73] is an indoor 802.11 positioning system. It exploits the idea of VOR (VHF Omnidirectional Ranging) using 802.11 hardware. Customized VOR base stations are equipped with directional antenna or phased antenna array to measure both angles and ranges. A host can measure its angle of arrival (AOA) of signals from a base station, relative to the center direction of the strongest signals. It can also estimate its range to a base station based on average signal strength (SS)

from all angles. A host determines its location based on AOA measurements (triangulation), ranges (trilateration), or a combination of both. VORBA also requires support from special infrastructure.

In [39], Haeberien *et al.* build a system using probabilistic techniques for localization across an office building. The whole building is divided into a set of cells with fixed sizes, typically one cell per office. In the first step to obtain training data, base station scans are conducted to cover entire area of each cell. Then for each base station, the distribution of its signal intensities in a cell is inferred by fitting the training data to a normal distribution. Finally, a probability distribution over cells is calculated to represent location estimation. Compared to this work, our approach derives probability distributions differently. Instead of relying on detailed signal intensity maps which are subject to unpredictable variations, our approach calculates probability distributions based on location constraints among nodes. Moreover, in our approach, the cell sizes are not fixed but adaptive to estimation confidence.

In [68], Madigan *et al.* develop an indoor positioning system based on Bayesian graphical models. It simultaneously locates a set of wireless clients (as opposed to localizing one user at a time). The system requires datasets of received signal strength (RSS) measurements from base stations to clients. The models capture the relationships between locations and RSS. The locations are inferred by applying Bayesian analysis to the models.

SeRLoc [60] is a range-independent localization algorithm suited for wireless sensor networks. The network consists of a small number of locators with

known locations and a large number of sensors to localize. Each locator is equipped with an array of sectored antennas. Sensors are equipped with omnidirectional antennas. Locators transmit beacon messages containing their coordinates and directions of antenna boundary lines. Beacons from different antennas cover different sectors. Given the locator-to-sensor communication range, a sensor can infer its location after receiving beacon messages. The final estimation is the center of gravity (CoG) of the intersection region of several sectors. SeRLoc uses a grid score table to find the intersection region.

A zero-configuration system is proposed in [67] for indoor localization. It takes on-line RSS measurements between 802.11 APs, which are used to analyze the effects of multi-path fading and environmental variations on RSS and create a mapping between RSS and geographical distance. A client measures RSS from APs and then estimates its distance to APs from the signal-distance map (SDM). A gradient descent method is applied to estimate location of the client that minimizes an objective function. The assumption behind this system is that, for an office building, RSS between APs and RSS between clients and APs have similar relationship with distances. This assumption may be problematic considering the different locations of clients and APs: APs are usually mounted on walls or other high places while clients are often on desks. There is normally more human mobility around a client than around an AP.

Thunder [107] is a centralized localization scheme for large outdoor WSNs. A centralized device broadcasts sound, together with radio signal containing its location. Each sensor can estimate its distance to the location of the centralized device

using Time Difference of Arrival (TDOA) approach upon receiving the sound and radio signal. The centralized device moves to three different noncollinear locations, so that each sensor obtains distances to three noncollinear points and localizes itself using trilateration.

Localization over multihop wireless connections: In MWNs, nodes are often multiple hops away from anchor nodes, therefore the location uncertainty is increased and localization is even more challenging. A number of interesting localization algorithms have been proposed for such networks. To avoid error accumulation, majority of the algorithms infer locations of nodes by formulating a constraint problem and solving it using some commonly used techniques. One key difference between the algorithms is how the location is represented, *e.g.*with a single point or a region. Location-dependent applications usually require single point representation as the final location estimation. But the intermediate estimations during localization process do not always have to be single points. Different algorithms may have different representations.

In [92], Savvides *et al.* develop a distributed localization approach that iterates through a two-phase process: ranging and estimation. During the ranging phase, each node estimates its distance to its neighbors, whereas during the estimation phase, nodes use the ranging information and their neighbors whose positions have been determined to estimate their own locations. In a followup work [93], the authors enhance the previous approach by formulating the problem as a global non-linear optimization problem. This limits error accumulation arising in [92].

In [96], Shang *et al.* propose to use multi-dimensional scaling (MDS) to determine location in a centralized fashion. MDS is a technique that can calculate coordinates from a matrix of pairwise distances. The localization accuracy is limited partly because it cannot handle violation of triangulation (especially for irregular-shaped networks). Later they develop a distributed MDS-based approach in [95]. It is shown to out-perform the centralized version in irregular-shaped networks by ignoring the distance information among nodes that are far apart.

In [17], Costa *et al.* introduce a distributed weighted-MDS algorithm, dwMDS, for localization in WSNs. It assigns larger weights to more accurate range measurements. Similar to [95], it essentially ignores distance estimation between out-of-range sensors by giving them 0 weight. In dwMDS, each node adaptively updates its location estimation by minimizing a local cost function.

In [9], Biswas *et al.* relax the localization problem to a semidefinite program (SDP) problem, and solve the problem using standard SDP techniques. To deal with noisy measurements, the authors develop two extensions to the basic SDP problem: a maximum likelihood based formulation and an interval based formulation.

In [71], Moore *et al.* present algorithms that use robust quadrilateral for localization. Their approach finds sets of four nodes that are fully connected, and localizes the fourth node based on the positions of the other three nodes. To prevent error accumulation, the four-node set needs to satisfy robust quadrilateral conditions. This improves accuracy at the cost of leaving some nodes unlocalized.

In [42], Hu *et al.* propose a sequential Monte Carlo localization (MCL)

method to enhance the accuracy of localization by exploiting mobility. In particular, the approach leverages mobility history to predict possible locations based on previous location samples and current movement, and uses the new connectivity information to eliminate inconsistent location samples.

In [91], Rudafshani *et al.* propose MSL and MSL* to improve and generalize MCL. The sampling procedure in MCL is modified to work in static networks. Each node only uses information from neighbors with better location estimates Convergence time and execution time of MSL and MSL* are faster than MCL, therefore the accuracy is improved for mobile networks.

In [65], Li *et al.* present REP for localization in anisotropic WSNs where holes exist. The basic idea is to better estimate distance between two sensors by rendering a shortest path around intermediate holes. Each sensor measures its distances to three seeds whose locations are known, then infers its location by trilateration.

Unlike most of the previous approaches, which represent inferred locations using points, [97] and [35] present approaches that represent locations as regions. In [97], the whole space is divided into a rectangular grid of small squares. Location estimates are represented by a set of squares calculated from location constraints. In [35], regions are represented with Bezier curves. Such a representation is shown to significantly improve accuracy. Motivated by these approaches, in this dissertation we also use region-based representation. Our approach is different in two ways. First, we represent regions with dynamic mesh cells with adaptive sizes. Second, we derive probability distribution over the cells. It achieves high accuracy and robustness without incurring significant computation cost.

Lastly, a set of probabilistic localization algorithms are presented in [77, 78, 85]. For each sensor in the network, they derive a probability distribution over the whole deployment area. Each position in the area is associated with a probability of the sensor being placed at that position. Eventually, the location of a sensor is estimated as the position with highest probability. These algorithms assume normal distribution of RSS or AOA measurements, and calculate probability distributions using distance or angle constraints. Different from them, our approach does not assume any distribution of RSS or AOA. In its basic form, it needs only connectivity information. It calculates probability distribution for discrete cells and reduces computation cost by adaptively controlling the cell sizes.

Analysis of limits on localization accuracy: In addition to developing novel localization algorithms, researchers have also analyzed the fundamental limits on localization algorithms. The limits can be about the accuracy of location estimations, or whether a node can even be localizable.

The authors in [23] compare a series of localization algorithms, and find that using commodity 802.11 technology over a range of algorithms, approaches and environments, it is expected to have a median localization error of 10 feet and 97th percentile error of 30 feet. They conclude that these limitations are fundamental and unlikely to be significantly improved without fundamentally more detailed environmental models or additional localization infrastructure. It points out that leveraging additional information is necessary in order to improve the accuracy.

In [33], Goldenberg *et al.* study partially localizable networks, in which

some nodes cannot be uniquely localized due to lack of location constraints. The authors identify a sufficient condition for a node to be uniquely localizable, and develop algorithms to determine which nodes are uniquely localizable and which are not.

Node placement: Complementary to localization algorithms, node placement algorithms have also been designed to reduce location ambiguity.

In [87] Ray *et al.* apply the theory of identifying codes to determine the placement of sensors so that each position is uniquely identified by a set of sensors that it can directly communicate with. The authors further extend their algorithms to tolerate errors (*e.g.*, sensor failures).

In [24], Eren *et al.* show that a network has a unique localization if and only if its corresponding grounded graph is generally globally rigid. Applying graph-rigidity literature, they develop approaches to constructing uniquely localizable networks, and study the computation complexity of localization. Node placement algorithms are complementary to localization algorithms. The localization algorithms should be applicable even when we do not have the flexibility to alter the graph to make it uniquely localizable.

2.3 Probabilistic Dynamic Mesh-Based Localization

As mentioned in the previous section, an important characteristic of various localization approaches is how the estimated location is represented. To achieve high accuracy and robustness, we adopt a region-based representation, where an

estimated location is represented as a region that consists of all points satisfying the location constraints extracted from the underlying network. We further improve the existing work [29,35] by deriving a probability distribution over the region to reflect the likelihood of the true position. Such probability distribution, combined with an explicitly represented region, provides much richer location information than a single position, and allows us to achieve higher accuracy in face of insufficient information and measurement errors.

Below we first present a probabilistic region-based localization approach. Then we describe two techniques to improve the efficiency of the approach. The first one combines multiple horizontal (or vertical) cells (in an estimated region) into a single segment, which reduces computation cost at the expense of slightly higher error. The second technique is based on a dynamic mesh, where mesh cells are dynamically adjusted according to the size and shape of the region. It can achieve both efficiency and accuracy.

2.3.1 Probabilistic Region-Based Localization

The probabilistic region-based localization proceeds as follows. First, every node's location is initialized to be the entire space. Then each node extracts location constraints by measuring the connectivity of the underlying network, and propagates these constraints to nodes within a certain hops away. (We use 3 hops in our evaluation.) If angle and received signal strength index (RSSI) measurements are available, they can be used to extract location constraints and processed in a similar way. Based on the constraints reported by other nodes and its own

observation, a node estimates its new location by pruning out the sub-regions that are inconsistent with the constraints. For the sub-regions that are consistent with the constraints, a node further computes a probability distribution over them. The approach is run in a distributed way.

Extracting location constraints: To estimate its location, a node first extracts location constraints from the underlying network. Examples of location constraints include "the distance between node i and node j is at most d" (also called distance constraints), and "the angle between line ij and the direction of North is within $[\theta_1, \theta_2]$" (also called angle constraints). Such location constraints can be obtained by measuring network connectivity and angle-of-arrival. In this section, we only consider distance constraints. We will consider angle constraints in Section 2.5.1.

To handle irregular wireless propagation, each wireless node is associated with two separate radii: R and r ($R \geq r$), where R denotes the maximum transmission range the node can reach, and r denotes the minimum transmission range the node can reach [35]. $R \neq r$ arises when the signal propagation is not the same in all directions. When node i can hear node j, we obtain a constraint: $d_{ij} \leq R_j$. This is a *positive constraint*. When node i cannot hear node j, we obtain a constraint: $d_{ij} > r_j$. This is a *negative constraint*.

Next we introduce some more notations. Let LC_{ji} denote a location constraint for node j using node i as a reference point. Let $POS()$ denote a positive constraint, and $NEG()$ denote a negative constraint. Let S_i and S_j be the estimated region of node i and j, respectively.

If node j can hear node i, we obtain a positive constraint: $d_{ij} \leq R$. Then the estimated region of node j can be expressed as:

$$S_j = POS(S_i, R) = \{p_j | \exists\, p_i \ \in S_i,\ d(p_i, p_j) \leq R\},$$

where $d(p_i, p_j)$ is the distance between two points p_i and p_j. This region is a union of discs that are centered at each point inside S_i with radius R. Similarly if node j cannot hear node i, we derive a negative constraint, and the region of node j is estimated to be

$$S_j = NEG(S_i, r) = \{p_j | \exists\, p_i \ \in S_i,\ d(p_i, p_j) > r\}.$$

If there are multiple constraints derived (*e.g.*, by using multiple reference points), the final output is the intersection of the regions from all these constraints. Note that while we use connectivity information to extract location constraints, our approach can easily incorporate other information, such as angle estimation and layout maps, which will be described in Section 2.5.1.

Computing probability: Next we describe how each node i derives a probability distribution P_i over its region S_i. To do so, we partition the whole space into (small) cells, where each cell is a square with a fixed size. A cell is the smallest unit for which we compute probability. Let s be a cell. $P_i(s)$ is the probability that node i is in s. Each location constraint gives a probability distribution over an estimated region. The final relative probability of each cell is the product of the probabilities derived from all constraints (including both positive and negative constraints). We further derive the absolute probability by normalizing the relative probabilities.

Below we show how to derive a probability distribution from one location constraint. Since the probability computation using positive and negative connectivity information is similar, we illustrate the idea by considering only a positive connectivity constraint.

First we describe how to compute probability $P_i(s)$ using an anchor node, a, whose location is known, as a reference point. Using network connectivity, we obtain a distance constraint from a to i as $d_{ia} \leq k * R$, where k is the number of hops between a and i. Therefore S_i is the disc centered at a with radius $k * R$. Since only connectivity information is available, we assume node i's location is uniformly distributed inside the circle. Therefore, for a cell g,

$$P_i(g) = \begin{cases} 0 & \text{if g is outside the circle,} \\ 1/c_1 & \text{otherwise,} \end{cases}$$

where c_1 is the number of cells inside the circle. (Note that application of negative connectivity information will change the above probability distribution. For example, if a node is 2 hop away from a, the fact that it is not a's immediate neighbor allows us to prune out the area of a circle centered at a with radius r.) To avoid leaving out the true position, a cell is considered "inside" the circle as long as it overlaps with the circle. Consequently, $S_i = \bigcup(g)$ is not exactly the region enclosed by the circle, but the union of all cells considered "inside" the circle. Therefore $1/c_1$ is an approximation since some cells are partially inside the circle. The accuracy of such approximation depends on the cell size. Smaller cell sizes reduce the approximation error at the cost of increasing computation and storage cost.

Next we describe how to compute probability $P_i(s)$ using a non-anchor node

(whose location is not known in advance) as a reference point. Consider a node i's neighbor j. For a cell $u_j \subset S_j$, the relative magnitude of its probability is determined by the probability of subregion in S_i that satisfies $d(u_i, u_j) \leq R$. This results in the following:

$$P_j(u_j) = \beta \cdot \frac{\sum_{u_i \subset d(u_i,u_j) \leq R} P_i(u_i)}{\sum_{u_i \subset S_i} P_i(u_i)} \quad (2.1)$$

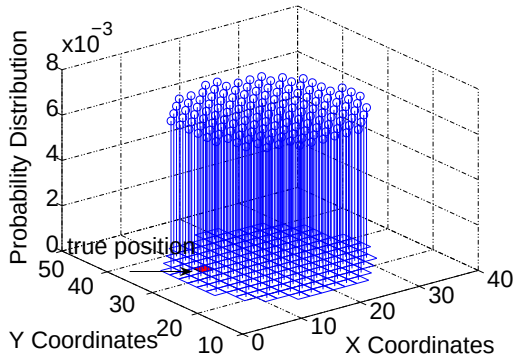
$$= \beta \cdot \sum_{u_i \subset d(u_i,u_j) \leq R} P_i(u_i) \quad (2.2)$$

where β is a normalization factor so that $\sum_{u_j} P_j(u_j) = 1$.

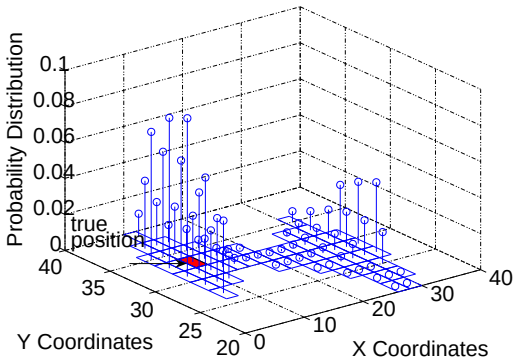
Figure 2.1 shows how a node's estimated location converges. After the first iteration, the region is approximately a circle since this node is a neighbor of an anchor. The probability distribution is uniform over all cells. After the second iteration, the estimated region is refined, with the updated probability distribution and smaller area, by leveraging the constraints from the anchors that are 2 hops away. After the third iteration, the region is reduced further (although the amount of reduction is less than in the second iteration because the constraints from the 3-hop neighbors have less impact on the region than constraints from the 2-hop neighbors). As it shows, the cell containing the true position (marked as the shaded cell) and its surrounding cells have significantly higher probabilities than the remaining region.

2.3.2 Enhancing Efficiency

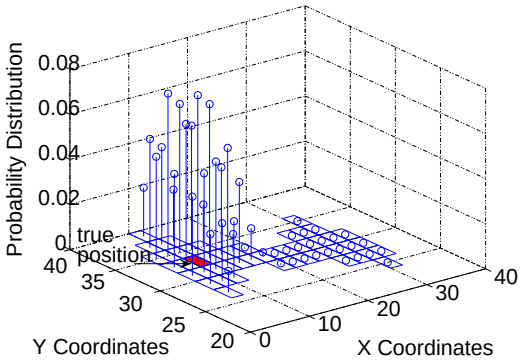
So far we consider using static grids. In this case, the computation cost is determined by the number of cells. If a node's location has high uncertainty due



(a) Snapshot after 1 iteration.



(b) Snapshot after 2 iterations.



(c) Snapshot after 3 iterations.

Figure 2.1: Snapshots of a node's estimated location for the first three iterations.

to lack of sufficient location constraints, its estimated region is large, resulting in a large number of cells and hence high computation and storage costs. In this section, we describe two techniques to improve the efficiency of the above localization approach. The first approach reduces the cost by combining horizontally (vertically) contiguous cells into a row (column) segment. The second approach dynamically adapts the cell size so that coarse-grained cells are used when the estimated region is large and fine-grained cells are used when the estimated region is small.

Segment-based localization: One way to reduce the complexity is to combine horizontally (vertically) contiguous cells into a row (column) segment. Since computation using row segments is similar as using column segments, in the following description we focus on using row segments. The width of each segment is fixed, but the length is variable. A row segment is specified by a 3-tuple, (y, x_1, x_2), where (x_1, y) is the left end and (x_2, y) is the right end. Each estimated region is represented as a set of row segments. We want to calculate the probability of each row segment containing the true position. Now the complexity is determined by the number of row segments.

Suppose we obtain node i's estimated region and the probability distribution over the region. We calculate its neighbor j's estimated region and probability distribution as follows. The location constraint LC_{ji} is $d_{ji} \leq R$. Hence, $S_j = POS(S_i, R)$. Let u_i denote a row segment of i, and u_j denote a row segment of j. The general formula to derive probability is similar to (2.1). Since a row segment may be significantly larger than a cell, treating partial overlap as complete

overlap may result in high error. Therefore we further calculate the fraction of a row segment that satisfies location constraints.

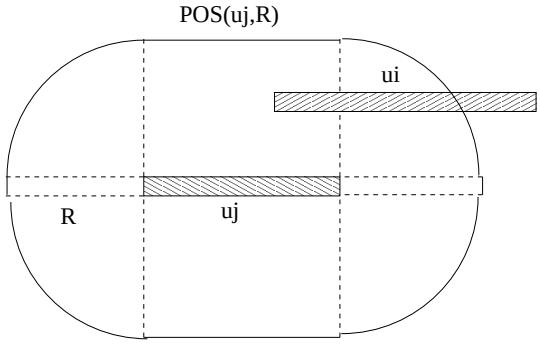


Figure 2.2: Example of Using Segments

Figure 2.2 shows an example. u_j is a row segment in S_j. $POS(u_j, R)$ is the region expanded from u_j by R. u_i is a row segment in S_i. u_i is partially in $POS(u_j, R)$. When calculating $P_j(u_j)$, we need to calculate the portion of u_i that is inside $POS(u_j, R)$.

Let $v_i = u_i \cap POS(u_j, R)$. Let $A(S)$ denote region S's size. Assuming uniform distribution within a segment, we have,

$$P_j(u_j) = \gamma \cdot \sum_{u_i \subset S_i} \frac{A(v_i)}{A(u_i)} \cdot P_i(u_i), \tag{2.3}$$

where γ is a normalization factor.

Probabilistic dynamic mesh-based localization (PDM): Combining consecutive cells in one dimension can significantly reduce computation and storage costs. On the other hand, its accuracy depends on how accurately a uniform distribution captures the actual probability distribution over the set of combined cells. When the

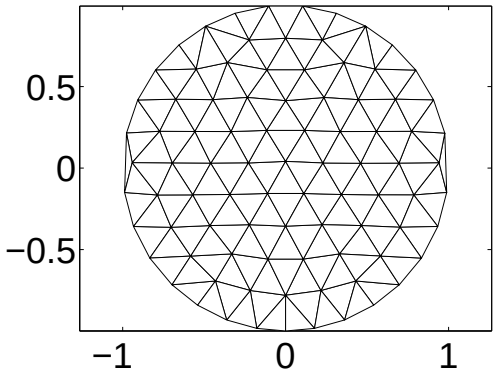
actual distribution significantly deviates from a uniform distribution, localization accuracy will decrease. To achieve both high accuracy and low cost, we propose an alternative approach that dynamically adjusts the cell size as needed.

At a high level, we use coarse-grained cells when the estimated region is large, and use fine-grained cells when the estimated region is small. To achieve this goal, we leverage mesh generation work developed in the area of computer graphics. We use DistMesh [21, 81] because it can efficiently generate high-quality meshes. DistMesh uses a *signed distance function* $d(x, y)$ to specify a region. The absolute value of $d(x, y)$ is the minimum distance from (x, y) to the boundary of the region, where a negative distance means it is inside the region and a positive distance means it is outside the region. It generates meshes using Delaunay triangulation, and optimizes node locations using a force-based smoothing procedure as described in [21, 81]. It also provides a parameter to control the sizes of triangles.

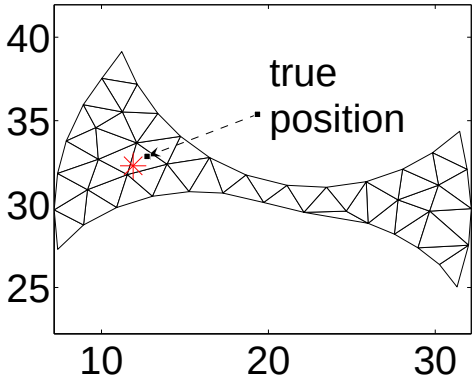
We apply DistMesh to localize wireless nodes as follows. Each node represents its estimated region using a set of triangular cells. A triangular cell is the smallest unit for which we compute a probability. We control the mesh structure so that each triangle has similar sizes in both dimensions, and the sizes of triangles are adaptive according to the size of the region. It is straightforward to write distance functions for distance constraints and angle constraints. Each node calculates its region based on the measured distance constraints. Given a combined distance function from all location constraints, DistMesh can generate a set of triangular meshes to represent the region that satisfies the location constraints.

Figure 2.3 illustrates two examples of triangular mesh generated by Distmesh.

Figure 2.3(a) shows the mesh cells for a circle. Figure 2.3(b) shows the mesh cell that represents the estimated region for the same node as in Figure 2.1, resulting from subtracting three circles from one circle.



(a) Mesh cells for a circle.



(b) Mesh cells for a node's location.

Figure 2.3: Triangular mesh generated by Distmesh.

After obtaining its estimated region, a node can derive the probability distribution over the triangles (inside the region) in a similar way as in static grids. Suppose we know the region and probability distribution over the triangles of a given node i. A neighbor j of node i has location constraint $d_{ji} \leq R$, and calculates

its region S_j as follows. Let t_i denote a triangle in S_i, and t_j denote a triangle in S_j. We derive the probability associated with t_j by first computing the fraction of t_i satisfying the location constraint, and then weighting the fraction by the probability of node i residing in t_i.

Figure 2.4 shows an example of deriving probability distribution. t_j is a triangle in S_j. $POS(t_j, R)$ is the region expanded from t_j by R. t_i is a triangle in S_i. t_i is partially in $POS(t_j, R)$. When calculating $P_j(t_j)$, we need to determine what fraction of t_i is inside $POS(t_j, R)$.

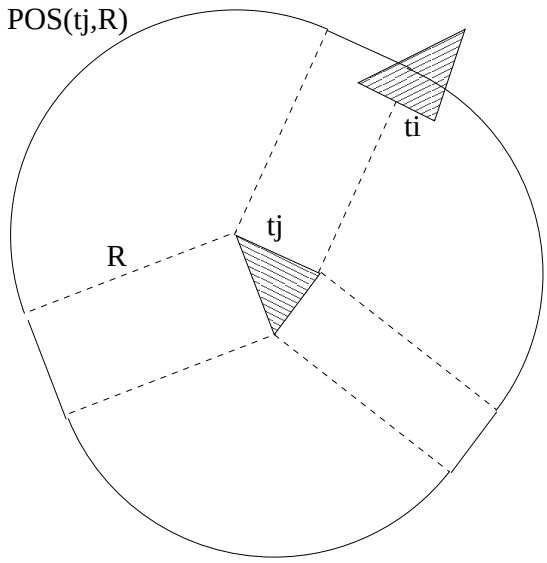


Figure 2.4: Example of mesh model.

Let $t_i' = t_i \cap POS(t_j, R)$. Assuming uniform distribution within a triangle, we have

$$P_j(t_j) \;=\; \gamma \cdot \sum_{t_i \subset S_i} \frac{A(t_i')}{A(t_i)} \cdot P_i(t_i), \tag{2.4}$$

where γ is a normalization factor.

2.4 Performance Evaluation

We evaluate localization schemes using a methodology similar to [96] and [95]. We uniformly place a set of nodes over a 2-dimensional space. We compare different localization schemes while varying the number of nodes (N), the maximum transmission range (R), and the fraction of anchor nodes (A). In this section, our evaluation uses one power level. In section 2.5.3.1, we will further study the effect of power control by varying the number of power levels.

We quantify the localization error using the same method as in [35]. For both Sextant and our approach, we use Monte Carlo sampling to sample 1000 points in a node's estimated region, and pick the one that minimizes the average error to other sampled points inside the region. The localization error is then calculated as the distance from this point to the node's true position.

However, there is a difference in choosing sample points between Sextant and our approach. Sextant uniformly samples points inside a region, whereas in our approach the number of sample points in a cell is proportional to its probability. As we will show, the probabilistic-based approach can significantly improve the localization accuracy.

Effects of the number of nodes Figure 2.5 shows the cumulative distribution of position errors for $N = 50$, $R = 12.5$, and $A = 10\%$. The size of the space is 50x50.

We make the following observations. First, PDM significantly out-performs Sextant. For example, the percentage of nodes achieving $\leq 30\% * R = 3.75$ errors

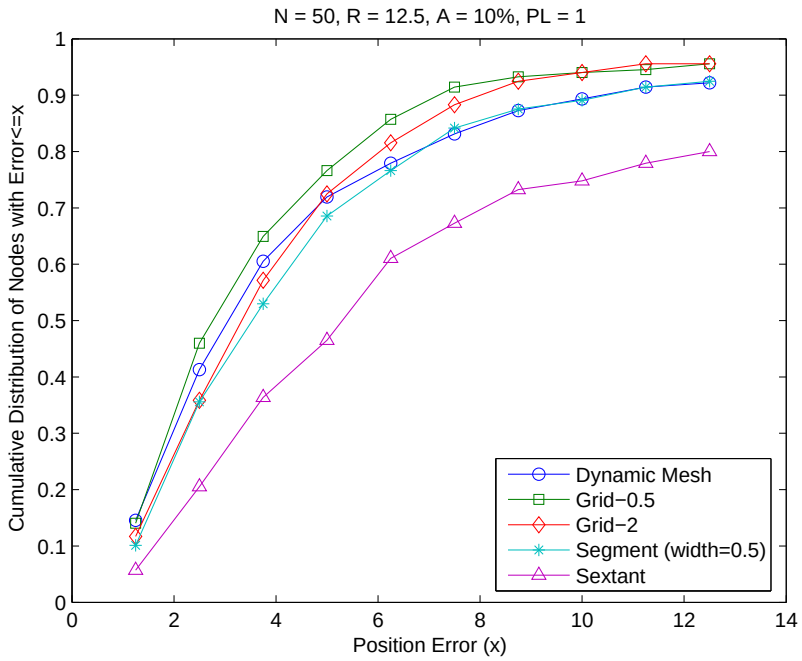


Figure 2.5: Probability distribution improves localization accuracy (50 nodes)

in dynamic mesh is 60% compared to 36% in Sextant. This is because in Sextant different points inside a region are treated equally, whereas PDM leverages the derived probability distribution over the region. Second, as we would expect, the static grid approach using 0.5x0.5 grids yields smaller errors than using 2x2 grids. Third, the dynamic mesh approach performs better than the static grid approach with 2x2 grids at the lower end of the errors ($\leq 10\%, 20\%, 30\% * R$ errors). Fourth, combining cells into segments with width 0.5 (denoted as "Segment (width=0.5)" in the figure) yields slightly larger errors than using static grids or dynamic meshes, but still out-performs Sextant by a significant amount.

Table 2.1 summarizes average running time of different algorithms. As we

Sextant	Grid-2	Grid-0.5	PDM	Segment
1.98	1.225	56.67	6.82	3.66

Table 2.1: Average running time in seconds using a 1200 MHz UltraSPARC-III+ processor with 16GB memory.

can see, the running time of static grid approach decreases with increasing grid size. When the grid size is as large as 2x2, the static grid approach takes less time than Sextant. In all other three schemes, the running time is longer than Sextant. (Note that Sextant code is from its original authors and it is implemented in JAVA, while all of our approaches are implemented in MATLAB. We expect the running time of our approaches can be significantly improved by converting the MATLAB code into C or JAVA.)

For the rest of evaluation, we choose PDM as a representative of our probabilistic approaches.

Figure 2.6 shows the performance for networks with 100 nodes in a space of size 70x70. Similar to networks with 50 nodes, PDM achieves higher accuracy than Sextant. For example, the percentage of nodes achieving $\leq 30\% * R = 3.75$ errors is 40% in Sextant, and is 67% in PDM. On average, Sextant takes 2.14 seconds per node to compute, and PDM takes 10.23 seconds per node to compute.

Effects of transmission range Transmission range R determines network density. More neighbors mean more location constraints, which usually result in higher localization accuracy. We vary R to obtain different network densities shown in Table 2.2. For simplicity, we assume the wireless propagation is regular (*i.e.* $R = r$)

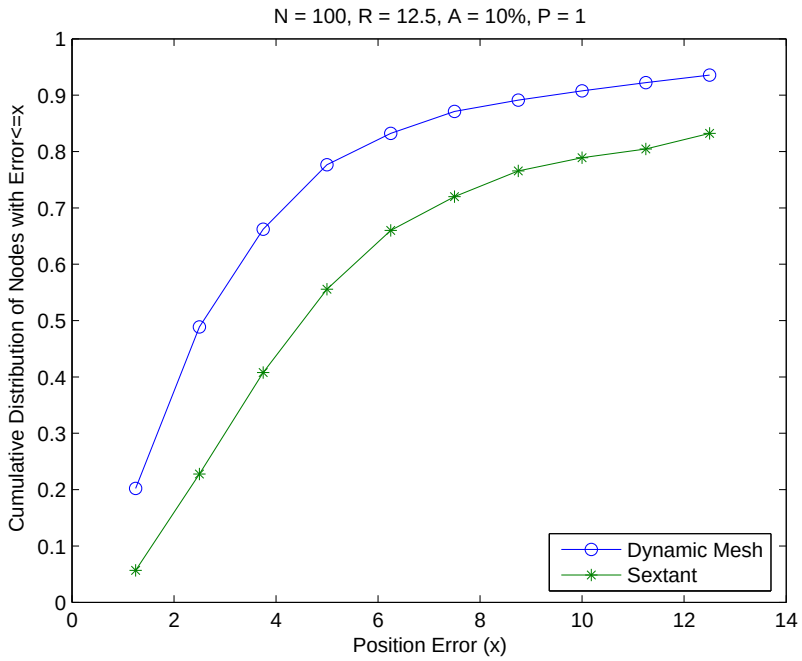


Figure 2.6: Probability distribution improves localization accuracy (100 nodes) in our simulation. It is not difficult to generalize to $R \neq r$ cases.

N	Space	$R = 10$	$R = 12.5$	$R = 15$
50	50x50	6.0612	8.9592	11.28
100	70x70	6.0562	8.58	11.28

Table 2.2: Average node degrees under different transmission ranges.

As described in [50], 6 is a "magic" average node degree for a wireless network to be connected. So we choose the shortest range to be 10, which gives an average node degree of 6.

Figure 2.7 shows the results for different transmission ranges, while fixing $A = 10\%$. The accuracy results of 50-node (not shown) is similar. Again, PDM consistently outperforms Sextant. As we would expect, the accuracy is higher when

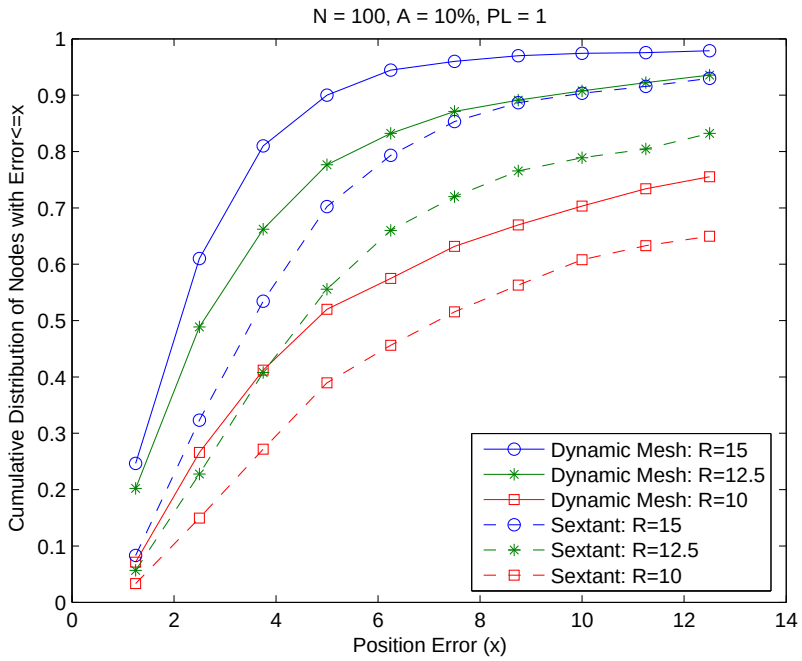


Figure 2.7: Effects of Transmission Range (100 nodes)

the transmission range is larger, which results in higher network density. Since the transmission range is determined by transmission power, there is a tradeoff between energy-efficiency and localization accuracy.

Effects of the fraction of anchor nodes Next, we study how the fraction of anchor nodes, A, affects localization accuracy. In our evaluation, $R = 12.5$. Figure 2.8 shows the localization accuracy of 100-node networks as we vary the anchor fraction from 5% to 20%. The results of 50-node networks are similar. As before, PDM yields lower error than Sextant. In addition, we find that the anchor fraction significantly affects localization accuracy. The more anchor nodes, the higher local-

ization accuracy. This is consistent with our expectation, because 1-hop neighbors of anchor nodes can be localized more accurately than nodes multiple hops away from anchor nodes due to smaller uncertainty. As shown in Figure 2.8, the increase in localization accuracy is significant as the anchor fraction increases from 5% to 10%. A further increase in the anchor fraction leads to more moderate increase in the accuracy. Therefore we use 10% as the anchor fraction for the remaining evaluation.

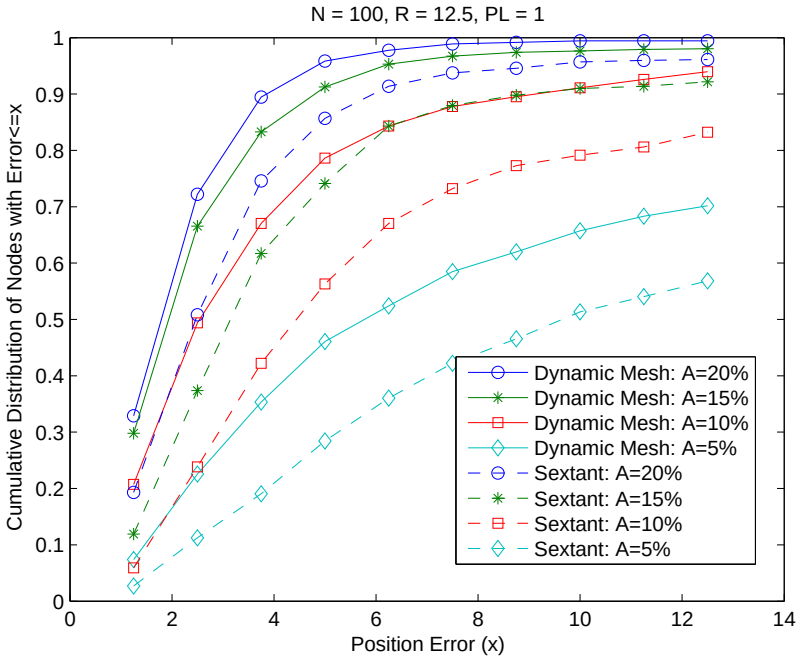


Figure 2.8: Effects of anchor nodes fraction (100 nodes)

Summary In this section, we compare different localization algorithms. Our results show that probabilistic region-based localization schemes using static grids,

dynamic meshes, and segments of grids, achieve higher localization accuracy than Sextant. In addition, PDM provides a reasonable balance between accuracy and computation cost.

2.5 Two Extensions

In this section, we extend our approach in two directions. First, to further improve the localization performance, we propose to extract and take advantage of additional information by (i) using power control, (ii) using carrier-sense range as another reference distance besides communication range, (iii) incorporating physical layout, and (iv) exploiting more powerful anchor nodes. The additional information is useful to many localization algorithms including ours. Second, we enhance the robustness of our approach against erroneous measurement by tolerating certain degree of inconsistency among location constraints. Finally, we evaluate the effectiveness of these extensions.

2.5.1 Extract and Leverage Additional Information

The accuracy of a localization system highly depends on the amount of information available. We propose several ways to obtain additional information. They can be used separately or jointly, and can be applied to different localization algorithms. Note that while this is not the first approach that uses additional information besides network connectivity to infer location, several of the techniques presented here are novel. In addition, we evaluate and compare the effects of the additional information.

Using power control: Power control enables wireless nodes to obtain additional information in the following way. Suppose each power level p_k has corresponding maximum and minimum transmission range $R(p_k)$ and $r(p_k)$. By adjusting the transmission power, if a node i finds out that it can communicate with another node j at power level p_k, but cannot communicate at power level p_{k-1}, the distance between i and j should be between $R(p_k)$ and $r(p_{k-1})$. This additional information makes range estimation more accurate, and can be easily incorporated into any localization algorithm. As we would expect, a larger number of power levels provides more information and improves localization accuracy. Power control is an interesting and practical way for obtaining additional information since power control is readily available in commercial wireless cards. In addition, it only requires nodes to obtain network connectivity information, and does not require signal strength measurements or additional hardware.

Using carrier-sense range: Many existing localization algorithms rely on network connectivity information for location estimation. This gives us information as to whether a node is within or outside the communication range of another node. However we do not have further information about the nodes that are outside the communication range.

We make an interesting observation: in addition to communication range, carrier-sense range can also be used as a reference for distance estimation. For example, if two nodes cannot sense each other's carrier, they are outside each other's carrier-sense range. This type of information is not available if we only use network connectivity, since the carrier-sense range is typically larger than the commu-

nication range. Let R and $R_{carrier}$ denote communication range and carrier-sense range, respectively. If two nodes are outside communication range but can sense each other's carrier, their distance should be within the range [R, $R_{carrier}$]; if two nodes cannot sense each other's carrier, their distance is larger than $R_{carrier}$.

To determine whether two nodes can sense each other's carrier, we can measure whether these nodes can simultaneously broadcast [2]. More specifically, we measure the broadcast rate from the two senders when they are active simultaneously, and denote it as $T_{together}$. We also measure the broadcast rate when the two senders are active separately, and denote it as $T_{separate}$. If $\frac{T_{together}}{T_{separate}}$ is close to 1, it means that the two nodes do not sense each other's carrier; otherwise they do.

As with power control, we extract more precise distance information using the carrier-sense range, and it can be applied to different localization schemes.

Using physical layout: In some applications, we may have a rough idea of physical layout of wireless nodes. For example, in residential mesh networks [89], we know that wireless nodes are deployed at different houses, and we also have a neighborhood layout map. The map provides additional information for us to narrow down the location. Since a node can only be located at one of the houses, its final estimated location should be the intersection of its estimated region (without considering the physical layout) and the regions occupied by the houses.

Using more powerful anchor nodes: As the previous work shows, angle information is valuable for location estimation. However, obtaining angle information often requires more expensive hardware (*e.g.*, directional antennas or additional

transmitters like ultrasound). In order to achieve both high accuracy and low cost, a promising approach is to use a combination of more powerful nodes and less powerful nodes. For example, only the anchor nodes are equipped with powerful devices for more detailed measurement, whereas the remaining nodes use cheap devices as usual. An interesting question is how much benefit such powerful anchor nodes offer. In this chapter, we study the following type of powerful anchor nodes: anchor nodes that are equipped with directional antennas for measuring angle information towards its immediate neighbors. We evaluate localization accuracy as we vary the fraction of anchor nodes.

2.5.2 Enhance Robustness

A node estimates its location by finding regions that satisfy a set of location constraints. Location constraints are usually obtained by measuring distances or angles between nodes. However, such measurements can be erroneous, and in some cases even lead to *inconsistent* location constraints. A set of location constraints are *inconsistent* if there is no point that can satisfy all these constraints.

We propose a technique on top of our probabilistic region-based approach to achieve robustness against inconsistent location constraints. We leverage the fact that majority of location measurements are consistent; and only a few constraints may contain significant errors and result in inconsistency. Therefore a mesh cell belongs to a node's estimated region as long as it satisfies most of the constraints. In our evaluation, we use 80% as a threshold (*i.e.*, a mesh cell is considered to belong to a node's estimated region if it satisfies at least 80% of the constraints for

that node). As part of our future work, we plan to choose the threshold adaptively.

Our robust localization proceeds in the following three steps. First, as before, every node propagates location constraints to all nodes within 3 hops away (*i.e.* TTL=3). Second, each node i calculates its own region based on the location constraints from other nodes. Location constraints from a node j determine a region S_{ij} for i. Unlike in Section 2.3, i does not calculate its region as $S_i = \cap_j S_{ij}$. Instead, S_i is calculated as the set of mesh cells u_i such that u_i satisfies at least 80% of the constraints. Finally, each node calculates the probability distribution over all mesh cells within its estimated region. This step is similar to what we describe in Section 2.3.

2.5.3 Performance Evaluation of Extensions

In this section, we evaluate the performance benefits of additional information and robustness enhancement.

2.5.3.1 Evaluation of Leveraging Additional Information

In this section, we study the effects of leveraging additional information. First, we examine the effect of power control by varying the number of power levels PL that a node can use for its transmission. Table 2.3 lists the transmission power at different levels, where P is the maximum transmission power. Note that $PL = 5$ corresponds to or approximates several commercial wireless cards (*e.g.*, Netgear WAG511 and Cisco Aironet 350 series). Next we examine the effect of carrier-sense range-based constraints by varying $R_{carrier} = 1.5R, 2R, 2.5R, 3R$, where

R is communication range. Table 2.4 summarizes the notation we use. Then we evaluate the performance benefit from incorporating a physical layout map. Finally, we examine the effect of using powerful anchors that have angle information. We consider three levels of angle measurement errors: large errors within $[-20, 20]$ degrees, medium errors within $[-10, 10]$ degrees and small errors within $[-5, 5]$ degrees. These values are consistent with commercial directional antennas.

PL	Fraction of maximum transmission power P
1	100%
2	25%,100%
3	6.25%,25%,100%
5	6.25%,12.5%,25%,50%,100%
10	6.25%,10%,12.5%,20%,25%,35%,50%,65%, 80%,100%

Table 2.3: Transmission power for different power levels

N	the number of nodes
R	transmission range
A	the fraction of anchor nodes
PL	the number of power levels
$R_{carrier}$	carrier-sense range

Table 2.4: Notation used in performance evaluation.

Effects of power control When only connectivity information is available, the distance measurement is binary–either $d \leq R$ or $d > R$. By adjusting the transmission power level, a node can extract more accurate distance constraints in the above form. As shown in Figure 2.9, the accuracy improves with an increasing number of power levels. For example, 20% nodes achieve position error within

$10\% * R = 1.25$ when 1 power level is used. In comparison, 32%, 35%, 50%, and 65% nodes achieve similar errors when the number of power levels is 2, 3, 5, and 10, respectively. This demonstrates that power control is effective in improving localization accuracy.

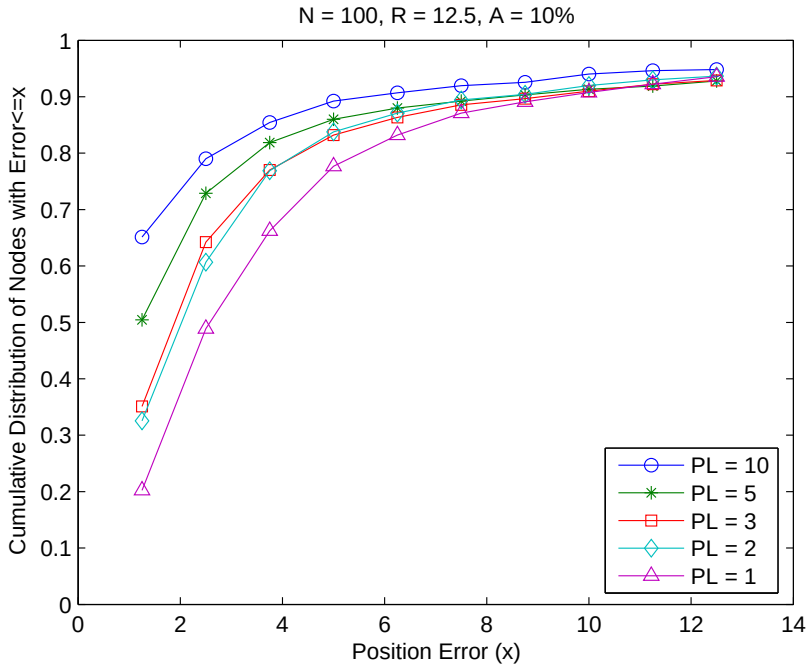


Figure 2.9: Effects of power control (100 nodes).

Effects of carrier sense constraint Besides power control, carrier-sense range can also help to extract more accurate distance constraints. As shown in Figure 2.10, compared with the base case without carrier sense information, constraints derived using carrier-sense ranges improve localization accuracy by a considerable amount. As the carrier-sense range increases, the negative constraints (*i.e.*, $d > R_{carrier}$)

become tighter, and the positive constraints (*i.e.*, $d < R_{carrier}$) become looser. Interestingly, $R_{carrier} = 2 * R$ yields the highest accuracy among all the carrier-sense ranges considered. This suggests that the positive and negative constraints extracted using $2 * R$ are especially effective under the scenarios we consider.

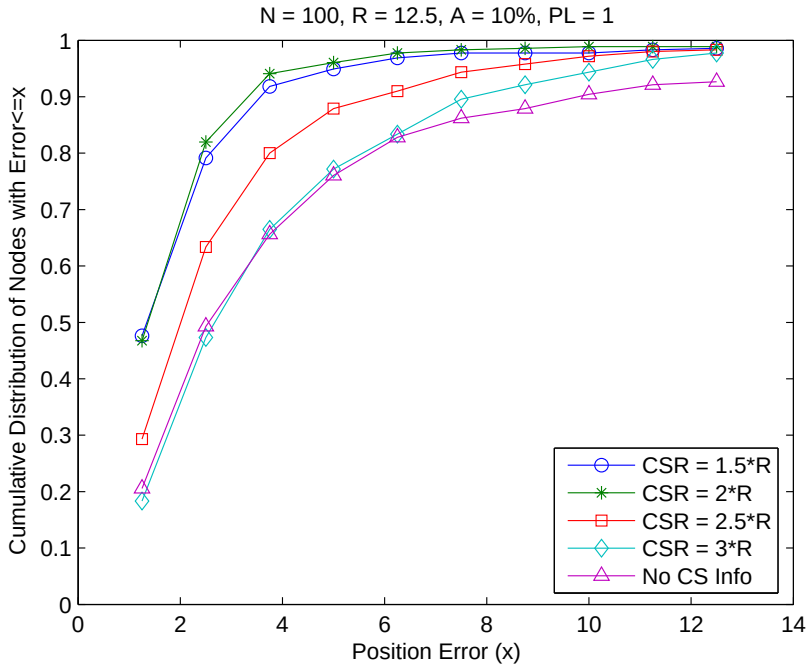


Figure 2.10: Effects of carrier sense constraints (100 nodes).

Effects of map constraint Next we study the performance gain from a layout map. In our evaluation, we obtain a real neighborhood map, which contains the coordinates of houses. We select 56 houses from the map over a 1400m x 700m space. Since there is no house size information, we generate the regions occupied by the houses as follows. Each house is a square and has the same size. A house is

centered at its coordinate, and its size, $hsize$, is determined based on the minimum distance between any pair of houses, d_{min}. In the localization process, each node derives its region and probability distribution based on the constraints imposed by the map (*i.e.*, a node can only be inside a house), as well as the location constraints from other nodes. We use transmission range of 150 meters, which gives an average node degree of 6.39.

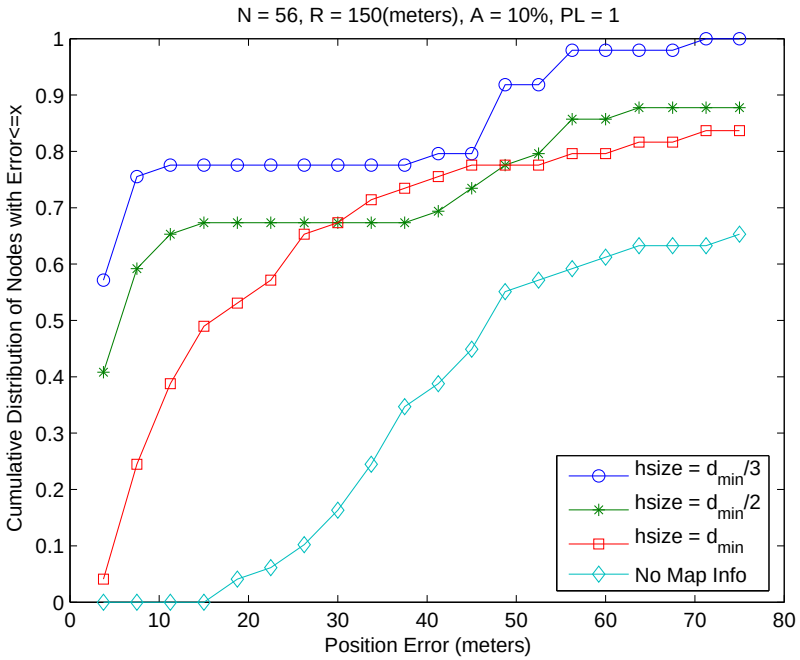


Figure 2.11: Effects of a physical layout.

As shown in Figure 2.11, a layout map significantly improves localization accuracy. In addition, the smaller house size, the higher localization accuracy. This is what we would expect. Because a node can only reside in a house, the location constraints imposed by the map is tighter for smaller houses. Nevertheless, even

when $hsize = d_{min}$, localization accuracy is still significantly higher than without the layout map.

Effects of powerful anchors Finally, we examine how anchor nodes with angle measurement affect the accuracy of localization. We use three levels of angle measurement errors: $[-20, 20]$ degrees, $[-10, 10]$ degrees and $[-5, 5]$ degrees. An estimated angle is then the true angle plus noise uniformly distributed within the error intervals. Figures 2.12 summarizes the results.

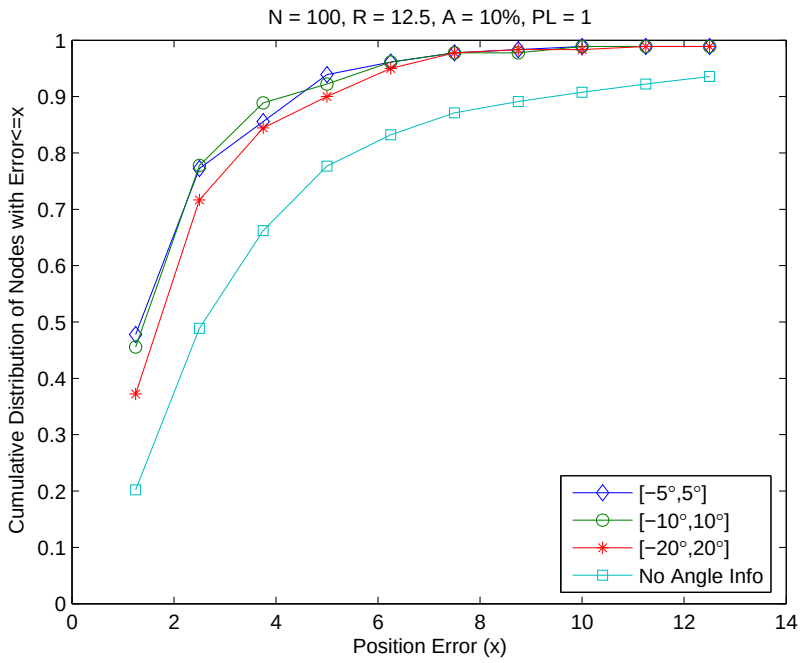


Figure 2.12: Effects of angle information (100 nodes).

We make the following observations. First, angle information helps to decrease the localization error significantly. Second, even when the angle mea-

surement contains errors of $[-20, 20]$ degrees, localization accuracy is still significantly higher than the accuracy achieved without angle information. Compared with $[-5, 5]$ degrees of angle measurement error, its accuracy is slightly lower at the low end of position errors, and comparable for the remaining position errors.

Summary In this section, we study the effect of additional information, including using power control, carrier-sense range-based constraints, a layout map, and angle measurements from anchor nodes. Our results demonstrate that the additional information is effective in significantly improving localization accuracy.

2.5.3.2 Evaluation of Robustness Enhancement

In this section, we evaluate the robustness of our extended localization algorithms. First we consider the case where the transmission range information is inaccurate. More specifically, each node's true communication range (R) is $R = R_{est} + R_{error}$, where R_{error} is a positive or negative range estimation error, and R_{est} is the communication range that we have estimated. R_{error} arises from the difference in transceivers' properties and environmental effects. While one may try to reduce R_{error} by individually calibrating each node (*e.g.*, obtaining conservative minimum and maximum communication ranges), such calibration is costly. Moreover even with calibration, errors cannot be completely eliminated due to changing environmental effects.

As shown in Figure 2.13, with robustness enhancement, the localization algorithm maintains high accuracy when the communication ranges contain up to

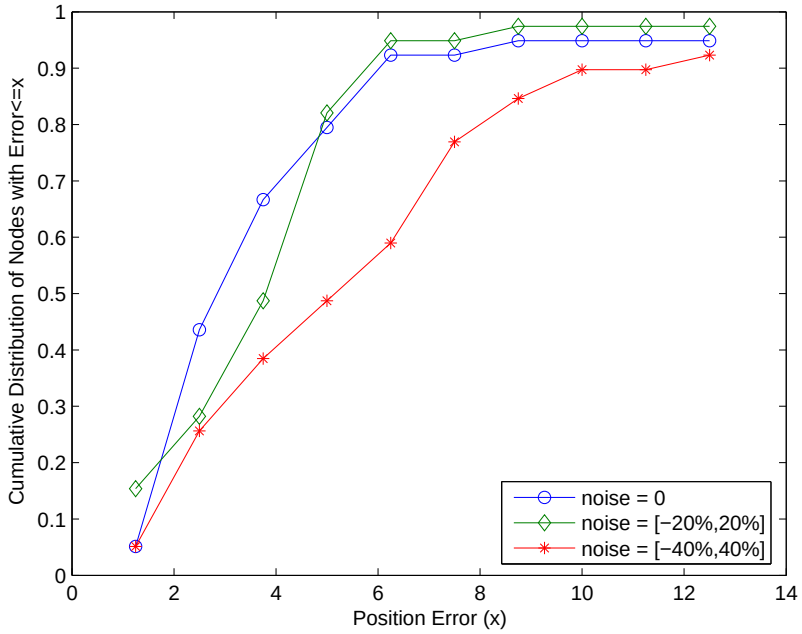


Figure 2.13: Effects of inaccurate communication range ($N = 50$, $R = 12.5$, $A = 10\%$, $PL = 1$).

$20\% * R$ errors. The accuracy is lower when R_{error} increases up to $40\% * R$, but still all nodes can be localized, with around 60% nodes achieving within $R/2 = 6.25$ position error.

Next we consider errors arising from malicious nodes. In our evaluation, we randomly select a few nodes as malicious nodes. Such a node pretends to be at a randomly generated location. It calculates a region of a circle centered at the false location with radius R, and then transmits this region as a false constraint to its neighbors. Figure 2.14 shows the effects of malicious nodes. There are two sets of curves, corresponding to the results of position errors within $R/2$ and within

R. "Grid-2" and "Robust Grid-2" curves represent the results from using fixed 2x2 rectangular cells with and without the additional robustness enhancement, respectively. After introducing such malicious nodes, not all nodes can be localized due to potentially inconsistent constraints. For the nodes that have inconsistent constraints and cannot be localized, their localization error is considered larger than R.

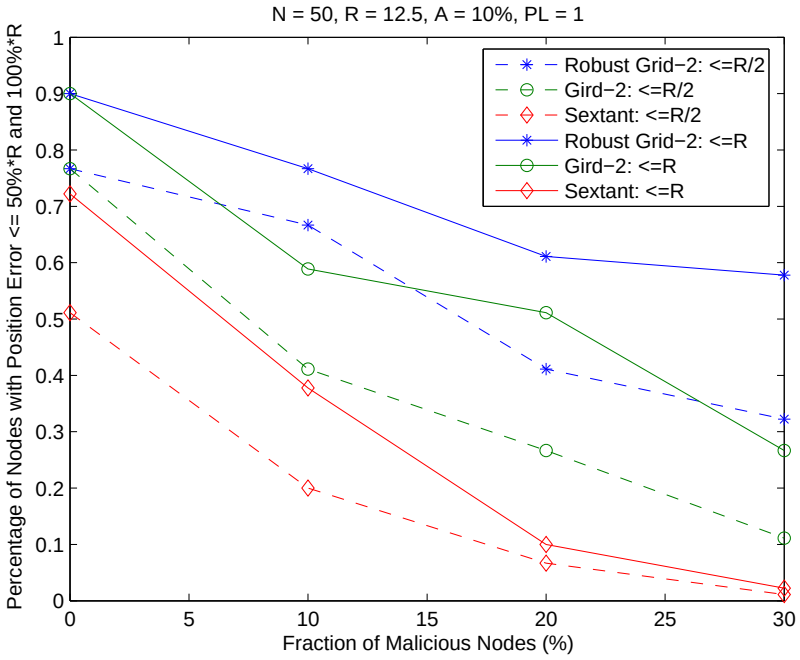


Figure 2.14: Effects of malicious nodes ($N = 50$, $R = 12.5$, $A = 10\%$, $PL = 1$).

As Figure 2.14 shows, even when the fraction of malicious nodes is only 10%, the percentage of nodes with position errors $\leq R/2 = 6.25$ drops as much as 30% under both Sextant and Grid-2. In comparison, with the additional robustness enhancement, the accuracy reduction under the "Robust Grid-2" is small especially when the fraction of malicious nodes is within 10% (only 10% reduction). More-

over, even when 30% nodes are malicious, majority of nodes can still be localized within errors of R under "Robust Grid-2". This demonstrates the effectiveness of our robustness enhancement.

Summary Our evaluation results show that the robustness enhancement is effective. The enhancement helps maintain high localization accuracy even when there are $20\% * R$ range errors or 10% malicious nodes.

Chapter 3

A General Model of Wireless Interference

3.1 Overview

Interference is fundamental to wireless networks. Due to the broadcast nature of the medium, transmissions from one sender interfere with the transmission and reception capabilities of other nodes. Understanding and managing interference is essential to the performance of wireless networks. For instance, it can directly benefit channel assignment [70, 90], transmit power control [47], routing [18, 22], transport protocols [64], and network diagnosis [15].

Unfortunately, the state of art in estimating the impact of interference is rather primitive. Much of the existing work is based on simple, abstract models of radio propagation (*e.g.,* the interference range is twice the communication range). While such models may predict the asymptotic behavior, they can be highly inaccurate in any given network [2, 53].

This has prompted researchers to devise models that are seeded using measurements from the underlying network [2, 88]. These measurements are usually collected in a simple configuration, such as each node sending by itself. They are then used to predict the impact of interference in more complex configurations such as multiple transmitting nodes. This is a promising direction because it makes no

assumptions about the nature of radio propagation which has proven difficult to model in real environments.

However, the existing measurement-based models are quite limited. They do not apply to configurations that have more than two senders or two flows, have unicast traffic, or have senders with finite demands. The only way today to predict network behavior under these general configurations is to actually measure it. (Indeed, most experimental research today is forced to adopt this methodology.)

But measurement alone is insufficient because it lacks predictive power and scalability. While it can accurately predict the performance of the measured configuration, it cannot predict performance for other configurations. To optimize network performance, one often needs to predict the performance of many alternative configurations. Since measuring all possible configurations is not feasible, it is necessary to develop a model to estimate network performance under arbitrary configurations (e.g., to perform what-if analysis).

In this chapter, we develop a general model of interference in heterogeneous multihop wireless networks with asymmetric link quality and non-binary interference relationships. Our model takes as input traffic demand and received signal strength (RSS) between pairs of nodes, which requires only $O(N)$ measurements in an N-node network. It then estimates the rate at which each sender will transmit and the rate at which each receiver will successfully receive packets.

Compared to existing measurement-based models [2, 88], we advance the state of art in three important ways. First, we go beyond the case of two senders

(or flows) to an arbitrary number of senders. This is challenging due to complex interactions among nodes. For instance, the sending rate of node s depends on those of all other nodes, which in turn depend on the sending rate of s itself. Second, we go beyond the case of broadcast transmissions. Unicast transmissions, which have both a data packet and an ACK, are more common in wireless networks. They introduces additional complexities that stem from retransmissions, exponential backoff, possibly asymmetric link qualities, and collisions not only among data packets but also due to ACK packets. Third, we go beyond the case of infinite traffic demands and model the more realistic case of finite demands. Most real networks have heterogeneous nodes with varying traffic demands.

Our model consists of three major components:

1. *An N-node Markov model for capturing interactions among an arbitrary number of senders.* This is based on a p-persistent CSMA approximation to the 802.11 distributed coordination function (DCF). It extends the previous models (*e.g.*, [8]) to multihop wireless networks, non-saturated demands, and asymmetric link quality.

2. *A receiver model of packet-level loss rates.* In particular, we find that slot-level and packet-level losses can be quite different depending on how losses are generated. Hidden terminals can significantly increase the packet-level loss rates well beyond the slot-level loss rates by spreading the lossy time slots across many packets. Based on this observation, our model captures both synchronized and unsynchronized packet-level collision losses.

3. *Unicast sender and receiver models.* We further extend the above broadcast sender and receiver models to capture interactions among unicast transmissions. We develop two major extensions for this purpose. The first extension models the retransmission and exponential backoff at the sender side, and the second extension models data/data, data/ACK, and ACK/ACK collision losses at the receiver side.

We evaluate our model using simulation as well as measurements over two wireless testbeds. Our results show that the model gives accurate prediction over a wide range of scenarios with both broadcast and unicast traffic, with both saturated and unsaturated demands, and across different number of senders. In simulation, where accurate RF profile is available, our model's root mean square error (RMSE) is less than 5% for both throughput and goodput predictions. In the testbeds, where RF profile is empirically measured and subject to measurement noise and bias due to lost packets, our model's RMSE is less than 12%. While our model is more general, we find that its accuracy is higher than the state-of-art model that considers the special case of two broadcast senders with infinite demands [88].

3.2 Related Work

Considerable research has been done in the area of modeling wireless networks. We broadly classify the existing work into three categories: (i) modeling single cell WLANs, (ii) general link throughput modeling, (iii) end-to-end throughput modeling.

Modeling single cell WLANs: The first category analyzes the performance of IEEE 802.11 Distributed Coordinated Function (DCF) [8, 58] for nodes all within communication range.

In [8], Bianchi *et al.* present an extremely simple Markov-chain model for performance analysis of DCF scheme. The model assumes ideal channel condition and finite number of hosts. It analyzes saturation throughput of the system. The key approximation is that, at each transmission attempt, packet collision occurs with constant and independent probability. Under this independence assumption, the model captures the bidimensional process of backoff time counter and backoff stage with a discrete-time Markov chain. Solving the Markov chain gives the stationary probability that a node transmits in a generic slot. The system throughput is then expressed as a function of this stationary probability.

In [58], Kumar *et al.* consider single cell WLANs where only one transmission is possible at any time. Their analysis begins with a similar key approximation as in [8], which leads to a fixed-point equation. They estimate the saturation throughput by solving the fixed-point problem.

While these models can estimate throughput under arbitrary numbers of senders, they assume packet transmissions to be synchronized and there are no partially overlapping transmissions. They do not apply to MWNs or multi-cell WLANs, where not all nodes can hear each other and partially overlapping transmissions are common. Moreover, they assume binary interference, which is not true in real wireless environments.

General link throughput modeling: The second category of work targets general topologies of MWNs or multi-cell WLANs where not all nodes are within communication ranges of each other. Because of the challenges presented by such topologies, existing models only handle restricted traffic scenarios.

Garetto *et al.* develop a two-flow model [31]. They focus on embedded subgraphs in an MWN, each of which consisting of four nodes and two flows. Depending on how the four nodes are connected with each other, there are totally twelve possible topologies. They develop models to predict each flow's short- and long-term performance for cases where senders do not hear each other, particularly an analytical model to characterize the long-term unfairness that arises in Asymmetric Incomplete State when 802.11 CSMA is used as the MAC protocol, and a model to capture the long-term fairness but short-term unfairness in Symmetric Incomplete State.

The work in [88] is closest to ours. Reis *et al.* develop a model to predict delivery rate and throughput under two competing broadcast senders. They first obtain an RF profile of an N-node network with $O(N)$ measurements, in which each node broadcasts in turn while other nodes record RSSI from the sender. The RF profile contains a mapping between delivery rate and RSSI for each receiver. Based on this profile, their model predicts active probabilities of each sender when two senders compete for the medium, and delivery rate at each receiver when both are active simultaneously. Their model is based on measurements from real network, therefore is more accurate than analytical models based on simplistic assumptions.

Our work falls into this category and advances the state-of-art by going

beyond pairwise interference and modeling interference among an arbitrary number of senders for both broadcast and unicast, both saturated and unsaturated demands.

In [46], Kashyap *et al.* independently present another measurement-based approach to modeling transmission capacity and link throughput in 802.11 networks. Their basic model supports arbitrary number of backlogged broadcast senders. This can be extended to support non-backlogged interferers and unicast. They evaluate and verify the cases of arbitrary number of backlogged broadcast interferers and single non-backlogged broadcast interferer. Similar to [88] and our work, they develop a MAC-layer model that is fed by a PHY-layer model. The PHY-layer model takes RSS measurements as profiling input and models deferral and packet capture. The sender-side of their MAC model is a discrete time Markov chain similar to that in [8]. They also assume a constant deferral probability in their model as in [8]. The receiver-side of their model considers both bit-error rate (BER) and SINR when estimating delivery rate. There are at least two major differences between their model and ours. First, their model requires a strong independence assumption, which does not hold in general. For example, their model assumes that two nodes carrier-sensing each other never transmit simultaneously, but in practice the collision probability is around 12% due to both nodes choosing same backoff timer. Our model can capture this nonnegligible event. Second, their model is based on non-linear constraints, hence is expensive to solve. They use either simulation which results in long running time or further approximation which leads to additional errors. In comparison, our model is based on linear constraints and can be solved analytically and efficiently without compromising accuracy.

End-to-end throughput modeling: The third category estimates the end-to-end throughput in multihop wireless networks [30, 37, 43, 59]. Since modeling end-to-end throughput is more difficult than one-hop throughput, to be tractable, such models only apply to specific scenarios. In particular, they either consider asymptotic behavior of wireless networks [37], or assume optimal scheduling [43, 59], or are limited to single flow scenarios [30].

In [37], Gupta *et al.* consider asymptotic behavior of wireless networks, in particular, the throughput capacity of networks. Through analytical modeling, they derive lower bounds on throughput capacity for both arbitrary and random networks.

In [43], Jain *et al.* investigate the impact of interference. They model the interference with a conflict graph and derive upper and lower bounds of optimal throughput for any given network and demand. The model assumes optimal scheduling by an omnipotent central entity, therefore it gives a best case bound. A similar problem is studied in [59], with the goal of designing provably good algorithms for arbitrary instances. The authors develop analytical models and distributed algorithms for joint routing and scheduling to achieve close to optimal throughput capacity.

Gao *et al.* present another model in [30] to compute end-to-end throughput capacity of a given flow. They first map an ad hoc wireless network into a contention graph to represent interference relationships. Then using an analytical model of 802.11 DCF, they derive channel idle and collision probabilities, and furthermore the final end-to-end throughput capacity.

While our work currently models link throughput, we believe it can be generalized to model end-to-end throughput in real networks if routing information is given. We will investigate this in the future.

3.3 Background on 802.11

The IEEE 802.11 standard [74] specifies two types of coordination functions for stations to access the wireless medium: distributed coordination function (DCF) and point coordination function (PCF). In this chapter, we focus on DCF, which is much more widely used than PCF. DCF is based on CSMA/CA. Before transmission, a station first checks to see if the medium is available by using virtual carrier-sensing and physical carrier-sensing. The medium is considered busy if either carrier-sensing indicates so. Virtual carrier-sensing considers medium is idle if the Network Allocation Vector (NAV) is zero, otherwise it considers the medium to be busy. Only when NAV is zero, physical carrier-sensing is performed. A station determines the channel to be idle when the total energy received at a node is less than the CCA (clear-channel assessment) threshold. In this case, a station may begin transmission using the following rule. If the medium has been idle for longer than a distributed inter-frame spacing time (DIFS) period, transmission can begin immediately. Otherwise, a station that has data to send first waits for DIFS and then waits for a random backoff interval uniformly chosen between $[0, CW_{min}]$, where CW_{min} is the minimum contention window. If at anytime during the period above the medium is sensed busy, the station freezes its counter and the countdown resumes when the medium becomes idle for DIFS. When the counter decrements to

zero, the node transmits the packet. In the case of unicast, if the receiver successfully receives the packet, it waits for a short inter-frame spacing time (SIFS) and then transmits an ACK frame. If the sender does not receive an ACK, it doubles its contention window to reduce its access rate. When the contention window reaches its maximum value, it stays at that value until a transmission succeeds, in which case the contention window is reset to CW_{min}.

3.4 Brief Description of Our Model

Our model takes traffic demands and RF profile as input and outputs the estimated sending and receiving rates for each node. Such a model is a powerful tool for performing what-if analysis and facilitating network optimization and diagnosis. More specifically, consider a network with N nodes. The inputs to the model are: $i)$ traffic demand from each sender s to each receiver r, and $ii)$ RF profile, which refers to the received signal strength (RSS) between every pair of nodes, denoted as R_{sr}. The outputs are the normalized throughput and goodput, denoted by t_{sr} and g_{sr}, respectively. t_{sr} is the rate at which s sends traffic to r and g_{sr} is the rate at which r receives successfully. Both t_{sr} and g_{sr} are normalized by the MAC-layer data rate.

In this dissertation, we focus on one-hop traffic demands, which means that traffic is only sent over one hop and not routed further. If r cannot hear from s, its receiving rate is zero. Modeling network performance under one-hop traffic demands is an important and necessary step towards estimating end-to-end throughput over multihop paths, which we plan to investigate in the future.

Our model operates as follow. First, we measure the RF profile of the network by letting each sender broadcast in turn and having the other nodes measure received RSSI values and loss rates. From these measurements, we recover pairwise RSS (R_{sr}) and background interference (B_r) due to sources other than nodes in the modeled network (Section 3.7). While we use custom traffic for our experiments, it may be feasible to perform these measurements using normal application traffic.

Then, we apply our *sender model* to estimate the amount of traffic sent by each sender under the given demand and our *receiver model* to estimate the amount of traffic successfully received. Our key contributions lie in the generality and accuracy of the sender and receiver models. They apply to both broadcast and unicast transmissions for an arbitrary number of senders, with and without saturated traffic demands. For saturated broadcast demands, our model can estimate throughput and goodput by computing the stationary probabilities of a Markov model. For unicast demands or unsaturated broadcast demands, the transition matrix of the Markov model involves additional variables and its stationary probabilities cannot be directly solved. Therefore we use an iterative framework, where we first initialize the variables in the transition matrix and then compute stationary probabilities, which are then used to update the transition matrix. Our results show that the iteration framework is effective and converges quickly (within 10 iterations in our evaluation).

We assume the following radio behavior. A transmitter s determines the channel is "clear" when the total energy it receives is below the CCA (clear-channel

assessment) threshold, β_s. A receiver r correctly decodes a transmission from a sender s when $i)$ its signal strength is at least radio sensitivity, γ_r; and $ii)$ the signal to interference-plus-noise ratio (SINR) is at least the SINR threshold, δ_r. We denote the thermal noise experienced by r as W_r. The values of β_s, γ_r, δ_r, and W_r are constant but radio-dependent.

The key notation used in this chapter is summarized in Table 3.1. We explain each term when it is first encountered.

	Model inputs: measured
R_{sr}	RSS from node s to r
B_r	Background interference at r
d_{sr}	Traffic demand from s to r
	Model inputs: radio-dependant constants
β_s	CCA threshold of s
γ_r	Radio sensitivity of r
δ_r	SINR threshold of r
W_r	Thermal noise of r
	Model outputs
t_{sr}	Normalized throughput: rate of traffic sent by s to r
g_{sr}	Normalized goodput: rate of traffic received by r from s
L_{sr}	Packet loss rate from s to r
	Other variables
S_i	Subset of nodes that are transmitting in state i
π_i	Probability that the network is in state i
M	Matrix of transition probabilities among states
$C(s\|S_i)$	Probability that channel is clear at s in state i
$Q(s)$	Probability for s to have data to send with backoff counter = 0
$\overline{OH}(s)$	Average overhead from DIFS, SIFS, and ACK at sender s
$\overline{CW}(s)$	Average congestion window of s
$T_\mu(s)$	Average packet transmission time for s

Table 3.1: A summary of key notation.

3.5 Broadcast Traffic

In this section, we present our model for broadcast traffic. Extensions to handle unicast traffic are presented in the next section.

3.5.1 Sender Model

The goal of the broadcast sender model is to estimate how much each sender can transmit given traffic demand. The classic Bianchi model [8] and its extensions (*e.g.*, [69]) model the behavior of 802.11 DCF by constructing a discrete Markov chain. To make the model tractable, all packet transmissions are assumed to be synchronized, *i.e.*, there will be no partially overlapping transmissions. In a general multihop wireless network, however, partially overlapping transmissions can be common because not all nodes can carrier sense each other. Thus, these models cannot be directly applied.

We develop a general N-node sender model based on Markov chains. We present it incrementally. First, we present the model for variable packet sizes and saturated traffic demands. Then, we extend it to handle fixed packet sizes and unsaturated demands in Sections 3.5.1.1 and 3.5.1.2. Finally, we describe techniques to enhance the scalability of the model in Section 3.5.1.3.

At a high level, we construct a Markov chain where each state i represents a set of nodes (denoted by S_i) that are transmitting in a time slot. Given N senders, the Markov chain has 2^N possible states (which we prune in Section 3.5.1.3). We derive the transition matrix M for the Markov chain based on 802.11 DCF and use it to compute the stationary probability π_i of each state. The throughput of node n

is then simply $t_n = \sum_{i|\, n \in S_i} \pi_i$.

Deriving the transition matrix M: In this section, we assume that nodes send variable-length packets with exponential distribution and that the state transitions of different nodes are independent. (We relax these assumptions in Section 3.5.1.1.) Because of independence, we can focus on computing the transition probabilities of an individual node, say n. This involves computing four transition probabilities for every state i: $i)$ staying in idle mode, $P_{00}(n|S_i)$; $ii)$ entering transmission mode, $P_{01}(n|S_i)$; $iii)$ exiting transmission mode, $P_{10}(n|S_i)$; and $iv)$ staying in transmission mode, $P_{11}(n|S_i)$. The probability $M(i,j)$ of the network transitioning from state i to j is:

$$\begin{aligned} M(i,j) = & \Pi_{n \in \overline{S_i} \cap \overline{S_j}} P_{00}(n|S_i) \times \\ & \Pi_{n \in \overline{S_i} \cap S_j} P_{01}(n|S_i) \times \\ & \Pi_{n \in S_i \cap \overline{S_j}} P_{10}(n|S_i) \times \\ & \Pi_{n \in S_i \cap S_j} P_{11}(n|S_i), \end{aligned} \tag{3.1}$$

where $\overline{S_i}$ denotes the complement of set S_i.

We compute the four per-node probabilities based on 802.11 DCF. A node can begin transmission when all the following three conditions hold: $i)$ its random backoff counter reaches 0; $ii)$ the medium is clear; and $iii)$ the node has data to

send. Thus:

$$\begin{aligned} P_{01}(n|S_i) &= Pr[medium\ is\ clear \wedge counter = 0 \wedge n\ has\ data] \\ &= Pr[medium\ is\ clear] \times Pr[counter = 0|medium\ is\ clear] \\ &\quad \times Pr[n\ has\ data|medium\ is\ clear \wedge counter = 0] \\ &= \frac{1}{\overline{CW}(n) + \overline{OH}(n)} \times C(n|S_i) \times Q(n), \end{aligned} \tag{3.2}$$

where $\overline{OH}(n)$ (for overhead) denotes the additional clear time slots that a node needs to wait in addition to $\overline{CW}(n)$, the average congestion window. For broadcast, $OH(n)$ is the DIFS duration in unit of time slots. $C(n|S_i)$ is the conditional clear probability and we compute it below. $Q(n)$ is the probability that n has data to send given that the medium is clear and the backoff counter is zero. For saturated demands, $Q(n) = 1$. We derive $Q(n)$ for unsaturated demands in Section 3.5.1.2.

For the staying idle probability, we have $P_{00}(n|S_i) = 1 - P_{01}(n|S_i)$.

To compute $P_{10}(n|S_i)$ and $P_{11}(n|S_i)$, assume that both transmission and idle times are exponentially distributed. (We relax this assumption in Section 3.5.1.1.) Let T_μ denote the average transmission time, computed based on packet size and transmission rate of the sender, and T_{slot} denote the duration of a time slot. Then we have:

$$P_{10}(n|S_i) = T_{slot}/T_\mu(n) \tag{3.3}$$

$$P_{10}(n|S_i) = 1 - T_{slot}/T_\mu(n) \tag{3.4}$$

The conditional clear probability $C(n|S) = \Pr\{I_{n|S} \leq \beta_n\}$, where $I_{n|S}$ is the total interference at n (when nodes in S are transmitting) and β_n is the CCA

threshold. $I_{n|S}$ is the sum of constant thermal noise W_n, the background interference B_n, and interference due to data transmissions by nodes in S except for n itself. Thus, $I_{n|S} = W_n + B_n + \sum_{s \in S \setminus \{n\}} R_{sn}$. To estimate this sum, we assume that each term is a lognormal random variable. The standard approach for dealing with the sum of such variables is to approximate it by a single lognormal random variable [25, 94]. Following Fenton [25], we find a lognormal random variable that matches the mean and the variance of $I_{n|S}$.

Formally, assuming that B_n and R_{sn} ($\forall s \in S$) are independent, we have $E[I_{n|S}] = W_n + \bar{\mathsf{B}}_n + \sum_{s \in S \setminus \{n\}} \bar{\mathsf{R}}_{sn}$, and $Var[I_{n|S}] = \mathsf{B}_n^{\text{var}} + \sum_{s \in S \setminus \{n\}} \mathsf{R}_{sn}^{\text{var}}$. Let e^Z be a lognormal random variable with $Z \sim N(\mu, \sigma^2)$. The first two moments of e^Z are $E[\mathsf{e}^Z] = \mathsf{e}^{\mu+\sigma^2/2}$ and $E[\mathsf{e}^{2Z}] = \mathsf{e}^{2\mu+2\sigma^2}$. Equating the first two moments of e^Z and $I_{n|S}$ gives: (i) $\mathsf{e}^{\mu+\sigma^2/2} = E[I_{n|S}]$, and (ii) $\mathsf{e}^{2\mu+2\sigma^2} = E[I_{n|S}^2] = Var[I_{n|S}] + (E[I_{n|S}])^2$. Therefore,

$$\mu = 2 \log E[I_{n|S}] - \frac{1}{2} \log E[I_{n|S}^2] \tag{3.5}$$

$$\sigma^2 = \log E[I_{n|S}^2] - 2 \log E[I_{n|S}] \tag{3.6}$$

We can then approximate $C(n|S)$ as

$$C(n|S) \approx \Pr\{\mathsf{e}^Z \leq \beta_n\} = \Pr\{Z \leq \log \beta_n\} = \Phi[\frac{\log \beta_n - \mu}{\sigma}],$$

where $\Phi[x] = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \mathsf{e}^{\frac{-u^2}{2}} du$ is the standard normal CDF.

Computing the stationary probabilities π_i: Having derived the transition matrix M, we can compute the stationary state probabilities π_i by solving the following

system of linear equations:

$$\sum_i \pi_i \, M(i,j) = \pi_j \quad (\forall j) \tag{3.7}$$

$$\sum_i \pi_i = 1 \tag{3.8}$$

where Equation (3.7) comes from the property that the stationary probabilities of the current and next states are equal, and Equation (3.8) normalizes the sum of the stationary probabilities to 1. For sparse matrix M, π can be efficiently solved, for instance, using *lsqr* [75]. Section 3.5.1.3 describes how to make M sparse to enhance scalability.

3.5.1.1 Handling Similar Packet Sizes

The previous section assumes variable packet sizes and independent transition probabilities for various nodes. But when all nodes use the same packet size, the independence assumption no longer holds. Specifically, when two nodes within carrier sense range transmit simultaneously, the start and end times of their transmissions will get synchronized with each other. The synchronization occurs because they will transmit simultaneously only when their random backoff counters both reach 0 within a tiny interval (otherwise the node that counts down to 0 later will sense the carrier and defer to the earlier transmission) [74]. The synchronized transmission and the same transmission time result in synchronized completion. Note that synchronization occurs even when the packet sizes are similar but not identical, since the transmissions start within one time slot difference and finish by a constant offset.

To handle such scenarios, we construct a *synchronization graph* for each set S of transmitting nodes as follows. Two nodes $s, t \in S$ are connected in the synchronization graph for S (denoted as $G_{\text{syn}}(S)$) if $C(s|\{t\}) < 0.1$ and $C(t|\{s\}) < 0.1$, where $C(i|\{j\})$ denotes the clear probability at node i when node j alone is transmitting. We find all the connected components in $G_{\text{syn}}(S)$, where each connected component represents a *synchronization group*.

Then we make two modifications to the transition probabilities $M(i, j)$ to account for synchronization effects. First, if there exists two nodes m and n in the same synchronization group of $G_{\text{syn}}(S_i)$ such that $m \in S_j$, $n \in \overline{S_j}$, then $M(i, j) = 0$. This is because all nodes in a synchronization group must exit the transmission mode together. Second, the probability for all nodes in a synchronization group to exit the transmission mode together is T_{slot}/T_{μ}. In comparison, with variable packet sizes, due to the independence assumption the above transition probability is $\Pi_n \frac{T_{slot}}{T_{\mu}(n)}$ (for all n in the synchronization group).

3.5.1.2 Handling Unsaturated Demands

The main challenge in handling unsaturated demands is estimating $Q(n)$ which is the probability that n has data to send when its backoff counter is 0 and the channel is clear at n. With saturated demands, it has a constant value of 1, but with unsaturated demands it must be computed to ensure that the traffic demands d_n are not exceeded. Computing $Q(n)$ is difficult due to strong inter-dependency among nodes. $Q(n)$ depends on how often the channel is clear at n, which depends on the amount of traffic generated by the other nodes, which in turn depends on $Q(n)$.

We develop an iterative algorithm to compute Q. The algorithm initializes Q to 1 for all senders. In each iteration, the algorithm first derives the transition matrix M based on the old Q values and computes the stationary probabilities π_i and the achieved throughput $t_n^{\text{old}} = \sum_{i:n\in S_i} \pi_i$. It then updates Q based on their values in the previous iteration. For this, we use the following relationships:

$$\frac{Q^{\text{old}}(n) \times T_\mu(n)}{Q^{\text{old}} \times T_\mu(n) + T_{\text{off}}(n)} = t_n^{\text{old}} \tag{3.9}$$

$$\frac{Q^{\text{new}}(n) \times T_\mu(n)}{Q^{\text{new}}(n) \times T_\mu(n) + T_{\text{off}}(n)} \leq d_n \tag{3.10}$$

$$Q^{\text{new}}(n) \leq 1 \tag{3.11}$$

where $T_{\text{off}}(n)$ represents the average time gap between two consecutive transmissions from n and $T_\mu(n)$ is the average transmission time of a packet from n.

Equation (3.9) captures the relationship between $Q(n)$ and the node's sending rate in the previous iteration. Equation (3.10) captures that the total amount of traffic sent by n cannot be more than the demand. Solving the three constraints yields

$$Q^{\text{new}}(n) = \min\left\{1,\ Q^{\text{old}}(n)\frac{d_n}{1-d_n}\frac{1-t_n^{\text{old}}}{t_n^{\text{old}}}\right\}. \tag{3.12}$$

At this point, we could directly use $Q^{\text{new}}(n)$ as our estimate for the next iteration. For quick convergence, we apply a relaxation procedure that is commonly used in equilibrium computation [54]. We set $Q^{\text{new}}(n)$ to be a linear combination of the computed $Q^{\text{new}}(n)$ and $Q^{\text{old}}(n)$: $Q^{\text{new}}(n) = \alpha \cdot Q^{\text{new}}(n) + (1-\alpha) \cdot Q^{\text{old}}(n)$. Our evaluation uses $\alpha = 0.9$, though we find that the model converges quickly for a wide range of α.

3.5.1.3 Enhancing Scalability

The general sender model, as presented earlier, requires 2^N states and $2^N \times 2^N$ transition matrix for N senders. To enhance scalability, we use two techniques that prune the states and transitions. First, we prune all those states that involve too many synchronized transmissions, which should occur with low probability. Specifically, given state i and the corresponding S_i, we eliminate i if the number of edges in the corresponding synchronization graph $G_{\text{syn}}(S_i)$ exceeds a given threshold, which is set to 2 in our evaluation. Second, we prune all those state transitions whose transition probabilities that are too low. Specifically, we reset the transition probability $M(i, j)$ to 0 if it falls below a threshold (which is set to 0.001 in our evaluation). With common configurations, transitions from i to j such that ≥ 2 synchronization groups exits the transmission mode is no longer allowed. Similarly, transitions with one node exiting transmission and one starting it tend to be filtered out as well. In this way, we can reduce the number of non-zero entries in M, thus improve the efficiency of sparse linear solvers such as *lsqr* in computing the stationary probabilities. The combination of these two techniques is highly effective. For example, consider 10 senders in a 5×5 grid topology, where any two direct horizontal, vertical, or diagonal neighbors can hear each other. Without pruning, the transition matrix has 1024 states and more than a million transitions. After pruning, it has only 370 states and 1736 transitions.

3.5.2 Receiver Model

We now present our receiver model for broadcast traffic. Our goal is to estimate the goodput g_{mn} (*i.e.*, the receiving rate). We have $g_{mn} = \eta t_m(1 - L_{mn})$, where L_{mn} is the packet loss rate from m to n, and $\eta = \frac{T_{\text{payload}}}{T_{\text{payload}}+T_{\text{header}}+T_{\text{preamble}}}$ represents the fraction of transmission time for the payload (excluding header and preamble overhead).

A key challenge in estimating L_{mn} is relating slot-level loss rates (derived from our *slot-level* Markov chain) to packet-level loss rates. Our experiments show that slot-level loss rates (*i.e.*, the fraction of time slots in which loss occurs) can be quite different from packet-level loss rates. For example, when loss comes from hidden terminals, where senders do not sense each other and cause collisions, a packet is usually corrupted partially. In this case, the packet-level loss rate can be significantly higher than slot-level loss rate. Consider transmission of 10 packets, which contain altogether 1000 time slots. Even if only around 10% slots (100 slots) are lossy, they can cause a packet loss rate as high as 100% if these lossy slots are distributed across all packets. Below we first analyze the slot-level loss rates and then convert them into packet-level loss rates.

3.5.2.1 Conditional Slot-Level Loss Probabilities

Let $I^*_{n|S} = W_n + B_n + \sum_{t\in S} R_{tn}$ be the total interference at n. We allow $t = n$ because n and sender m may be transmitting at the same time. At the slot level, a loss occurs when either SINR falls below δ_n and or RSS falls below γ_n. Let $\ell_{mn|S} = \Pr\{\frac{R_{mn}}{I^*_{n|S}} < \delta_n\}$ be the slot-level loss rate caused by low SINR when S is

transmitting. Let $\ell_{mn}^{\text{rss}} = \Pr\{R_{mn} < \gamma_n\}$ be the slot-level loss rate caused by low RSS.

$\ell_{mn|S}$ can be computed in a manner similar to that of conditional clear probability. Approximate $I_{n|S}^* = W_n + B_n + \sum_{t \in S} R_{tn}$ with a single moment-matching lognormal random variable e^Z, where $Z \sim N(\mu, \sigma^2)$. Since the ratio of two independent lognormal random variables R_{mn} and e^Z is also a lognormal random variable, let $\mathsf{e}^{Z'} = \frac{R_{mn}}{\mathsf{e}^Z}$, where $Z' \sim N(\mu', \sigma'^2)$. We have $\mu' = E[\log R_{mn}] - \mu$ and $\sigma'^2 = Var[\log R_{mn}] + \sigma^2$. Thus:

$$\ell_{mn|S} = \Pr\left\{\frac{R_{mn}}{I_{n|S}^*} < \delta_n\right\} \approx \Pr\{\mathsf{e}^{Z'} < \delta_n\} = \Phi\left[\frac{\log \delta_n - \mu'}{\sigma'}\right]. \tag{3.13}$$

There are two ways to estimate $\ell_{mn|S}^{\text{rss}} = \Pr\{R_{mn} < \gamma_n\}$. When the distribution of R_{mn} is available, we can directly compute $\Pr\{R_{mn} < \gamma_n\}$. In practice, R_{mn} has to be estimated and is subject to estimation error. To minimize error, we observe that when there is only a single sender and no external interference, all losses are due to low RSS. Thus we can directly use the measured packet loss rate under transmissions from a single sender m to estimate ℓ_{mn}^{rss}.

3.5.2.2 Packet-Level Loss Probability L_{mn}

Packet losses can be broadly divided into three categories. First, packet-level losses can stem from low RSS (L_{mn}^{rss}), which is not directly related to collision. Second, losses can stem from collision with packets from the same synchronization group. In this case, the fraction of lost packets (L_{mn}^{syn}) is close to the fraction of lost slots. Third, losses can also stem from asynchronous transmissions (*e.g.*, from

hidden terminals). In this case, the packet level loss rate (L_{mn}^{asyn}) can be much higher than the slot-level loss rate. Assuming independence among the three types of losses, the overall packet-level loss rate is:

$$L_{mn} = 1 - (1 - L_{mn}^{\text{rss}}) \times (1 - L_{mn}^{\text{syn}}) \times (1 - L_{mn}^{\text{asyn}}) \tag{3.14}$$

L_{mn}^{rss} can be estimated as $1 - (1 - \ell_{mn}^{\text{rss}})^{T_\mu / T_{slot}}$. Note that when measured packet loss rate from a single sender m is available, we can directly use the loss rate as L_{mn}^{rss} without first converting it into ℓ_{mn}^{rss}.

To derive L_{mn}^{syn}, let $SS(m)$ be the set of states that contain at least one synchronized transmissions involving m, *i.e.*,

$$SS(m) \triangleq \{i \mid m \in S_i \wedge S_i \text{ contains node(s) synchronized with } m\}.$$

We then estimate L_{mn}^{syn} as

$$L_{mn}^{\text{syn}} = \frac{\sum_{i \in SS(m)} \pi_i \ell_{mn|S_i}}{\sum_{i|\, m \in S_i} \pi_i} = \frac{\sum_{i \in SS(m)} \pi_i \ell_{mn|S_i}}{t_m},$$

which gives the total fraction of slot-level losses occurred when m collides with background traffic synchronously. Recall that t_m is the throughput of m.

To derive L_{mn}^{asyn}, we first compute the slot-level loss rates due to asynchronous collisions between foreground and background traffic:

$$\ell_{mn}^{\text{asyn}} = \frac{\sum_{i: i \notin SS(m) \wedge m \in S_i} \pi_i \ell_{mn|S_i}}{\sum_{i|\, m \in S_i} \pi_i} = \frac{\sum_{i: i \notin SS(m) \wedge m \in S_i} \pi_i \ell_{mn|S_i}}{t_m}$$

We model the background traffic as an ON/OFF process with exponentially distributed ON and OFF periods. Let $\overline{t_{\text{on}}^{\text{bg}}}$ and $\overline{t_{\text{off}}^{\text{bg}}}$ denote average durations of the two

periods. Under the assumption that the foreground and background traffic arrives independent of each other, the slot-level loss rate experienced by the foreground traffic should be equal to the fraction of time that the background traffic is in ON periods. That is,

$$\frac{\overline{t_{\text{on}}^{\text{bg}}}}{\overline{t_{\text{on}}^{\text{bg}}} + \overline{t_{\text{off}}^{\text{bg}}}} = \ell_{mn}^{\text{asyn}} \tag{3.15}$$

We have $\overline{t_{\text{on}}^{\text{bg}}} = T_\mu$ and the above equation yields $\overline{t_{\text{off}}^{\text{bg}}} = \frac{1-\ell_{mn}^{\text{asyn}}}{\ell_{mn}^{\text{asyn}}} T_\mu$.

A packet is successfully received if it starts with the background OFF period and the rest of this OFF period lasts at least the packet transmission time. We thus have:

$$1 - L_{mn}^{\text{asyn}} = \frac{\overline{t_{\text{off}}^{\text{bg}}}}{\overline{t_{\text{off}}^{\text{bg}}} + T_\mu} \cdot \exp\left[-\frac{T_\mu}{\overline{t_{\text{off}}^{\text{bg}}}}\right] \tag{3.16}$$

$$= (1 - \ell_{mn}^{\text{asyn}}) \cdot \exp\left[-\frac{\ell_{mn}^{\text{asyn}}}{1 - \ell_{mn}^{\text{asyn}}}\right] \tag{3.17}$$

where the first term on the right hand side of Equation (3.16) is the probability that the packet transmission starts in the OFF period and the second term is the probability that the rest of this OFF period lasts for at least T_μ.

3.6 Unicast Traffic

In this section, we extend our broadcast models to handle unicast traffic. There are two key differences between unicast and broadcast transmissions. On the sender side, the transition matrix M is different under unicast due to additional ACK overhead and exponential backoff. On the receiver side, there are additional

losses due to ACKs colliding with both data and other ACKs. We present the sender side extensions followed by the receiver side extensions.

3.6.1 Extensions to Sender Model

The transition matrix for the sender, in particular, $\overline{OH}(m)$ and $\overline{CW}(m)$ in Equation (3.2) are different for unicast traffic. Unicast has additional overhead from SIFS and ACK. If t_{SIFS} and t_{ACK} denote the number of time slots for SIFS and ACK, $OH_{mn} = t_{DIFS} + t_{SIFS} + (1 - L_{mn})t_{ACK}$.

$\overline{CW}(m)$ can be derived based on the packet-level loss rate L_{mn} across all receivers as follows. Let $H(L)$ be the average contention window under packet loss rate L, and $RMAX$ be the maximum number of retransmissions. Then:

$$H(L) = \sum_{i=0}^{RMAX} \frac{\min\{(CW_{\min} + 1)2^i - 1, CW_{max}\}}{2} L^i \tag{3.18}$$

A sender may transmit to more than one receiver, each with a different loss rate. We estimate $\overline{OH}(m)$ and $\overline{CW}(m)$ as the weighted average over all receivers, where the weights are based on the total transmissions to the receivers. Let G_{mn} denote the expected number of transmissions (including the first transmission) for each data packet sent from m to n. For simplicity, the weight can be approximated as $\frac{G_{mn} \times d_{mn}}{\sum_r G_{mr} \times d_{mr}}$. Assuming independent packet losses, $G_{mn} = \sum_{i=0}^{RMAX} \ell_{mn}^i$. Therefore:

$$\overline{CW}(m) = \sum_{n \in Recv(m)} H(L_{mn}) \frac{G_{mn} \times d_{mn}}{\sum_{r \in Recv(m)} G_{mr} \times d_{mr}} \tag{3.19}$$

$$\overline{OH}(m) = \sum_{n \in Recv(m)} OH_{mn} \frac{G_{mn} \times d_{mn}}{\sum_{r \in Recv(m)} G_{mr} \times d_{mr}} \tag{3.20}$$

where $Recv(m)$ denotes m's receivers.

The new expressions of $\overline{OH}(m)$ and $\overline{CW}(m)$ above enable us to compute the transition matrix and state probabilities π_i for unicast traffic. From that the throughput from m to n can be computed as $t_{m*} = \sum_{i|m \in S_i} \pi_i$ and $t_{mn} = t_{m*} \cdot \frac{d_{mn} \times G_{mn}}{\sum_r d_{mr} \times G_{mr}}$.

3.6.2 Extensions to Receiver Model

Consider node m sending data to node n. As for broadcast, we decompose packet-level unicast loss rate L_{mn} into three components: (i) L_{mn}^{rss} – losses due to low RSS (and not collisions), (ii) L_{mn}^{syn} – losses due to synchronized collisions between foreground and background traffic, and (iii) L_{mn}^{asyn} – losses due to asynchronous collisions between foreground and background traffic. Assuming independence again, $L_{mn} = 1 - (1 - L_{mn}^{\text{rss}}) \times (1 - L_{mn}^{\text{syn}}) \times (1 - L_{mn}^{\text{asyn}})$.

The key extensions that we make are: (i) extend L_{mn}^{rss} to include RSS induced losses for both DATA and ACK packets, and (ii) extend $\ell_{mn|S}$ to include SINR induced losses due to collisions between ACK/DATA, DATA/ACK, ACK/ACK (in addition to DATA/DATA), which is then used to compute L_{mn}^{syn} and L_{mn}^{asyn} in the same way as for broadcast. Below we describe these extensions in detail.

Estimating RSS-induced loss L_{mn}^{rss}: As before, let $\ell_{mn}^{\text{rss}} = \Pr\{R_{mn} < \gamma_n\}$. Similarly, let $\ell_{nm}^{\text{rss}} = \Pr\{R_{nm} < \gamma_m\}$. The combined RSS-induced loss on DATA and ACK is then

$$L_{mn}^{\text{rss}} = 1 - (1 - \ell_{mn}^{\text{rss}})^{T_\mu(m)/T_{slot}} \times (1 - \ell_{nm}^{\text{rss}})^{T_{ACK}(n)/T_{slot}}$$

where $T_{ACK}(n)$ is the duration of an ACK sent by n. Here we assume that the RSS-induced losses for DATA and ACK are independent.

Estimating SINR-induced loss $\ell_{mn|S_i}$: We consider the following three cases of low SINR induced losses:

C1. *DATA loss caused by other DATA's:* DATA transmissions from m to n get lost due to collisions with DATA transmissions by nodes in $S_i \setminus \{m\}$. This case is already considered in the broadcast scenario and we have $\ell_{mn|S_i}^{\text{C1}} = \Pr\{\frac{R_{mn}}{I_{n|S_i}^{C1}} < \delta_n\}$, where $I_{n|S_i}^{C1} = W_n + B_n + \sum_{t \in S_i} R_{tn}$.

C2. *DATA loss caused by other DATA's and ACK's:* A synchronization group G ($m \notin G$) of $G_{\text{syn}}(S_i)$ exits the transmission mode while all nodes in $S_i \setminus G$ continues transmitting. In this case, the ACK's generated by recipients of nodes in G could potentially increase the data loss rate from m to n. To quantify such effects, let random variable $R_{\text{ack}}(m, n)$ denote the noise experienced by n when m stops transmitting, causing some node in $Recv(m)$ to send an ACK back to s. The total noise at n is therefore $I_{n|S_i}^{C2} = W_n + B_n + \sum_{t \in S_i \setminus G} R_{tn} + \sum_{t \in G} R_{\text{ack}}(t, n)$. Let j be the new state after nodes in G stop transmitting. Among all possible next states for i to transit to

(excluding i itself), j will be chosen with probability $\frac{M(i,j)}{1-M(i,i)}$. Therefore, the total slot-level loss rate caused by ACK's is $\ell^{\mathrm{C2}}_{mn|S_i}(G) = \frac{M(i,j)}{1-M(i,i)}\Pr\{\frac{R_{mn}}{I^{C2}_{n|S_i}} < \delta_n\}$.

C3. *ACK loss caused by other DATA's and ACK's:* The synchronization group G' that m belongs to exits the transmission mode while all nodes in $S_i \setminus G'$ continue transmitting. Let j' be the new state resulted from such transition. During such transition, ACK's sent by receivers of nodes in $G' \setminus \{m\}$ combined with DATA's sent by nodes in $S_i \setminus G'$ together can potentially corrupt ACK's sent from n to m. The probability for this to occur is $\ell^{\mathrm{C3}}_{mn|S_i} = \frac{M(i,j')}{1-M(i,i)}\Pr\{\frac{R_{nm}}{I^{C3}_{m|S_i}} < \delta_n\}$, $I^{C3}_{m|S_i} = W_m + B_m + \sum_{t\in S_i\setminus G'} R_{tm} + \sum_{t\in G'\setminus\{m\}} R_{\mathrm{ack}}(t,m)$.

Note that with our pruning strategies described in Section 3.5.1.3, when a group G stops transmitting, we do not need to worry about having another group entering or exiting the transmission mode at the same time (because the transition probability would become too small). Under the independence assumption, we can compute the combined conditional slot-level loss rate as

$$1 - \ell_{mn|S} = (1 - \ell^{\mathrm{C1}}_{mn|S}) \times \Pi_{G:m\notin G}(1 - \ell^{\mathrm{C2}}_{mn|S}(G)) \times (1 - \ell^{\mathrm{C3}}_{mn|S}) \qquad (3.21)$$

Now the only remaining issue is to estimate $R_{\mathrm{ack}}(m,n)$. The main challenge is that m may have multiple receivers and RSS from different receivers are different. To address this, for each sender we compute the weighted average of interference that its receivers generate, where the weights are based on the traffic

demands and delivery probabilities to the receivers. Specifically, we approximate RSS contributed by m's ACKs at node n as a log-normal distribution by computing its mean and variance, denoted by $\bar{R}_{ack}(m,n)$ and $R^{var}_{ack}(m,n)$, as

$$\bar{R}_{ack}(m,n) = \sum_{r \in Recv(m)} \frac{G_{mr} d_{mr}}{\sum_{r'} G_{mr'} d_{mr'}} \cdot (1 - L_{mr}) \cdot \bar{R}_{rn} \tag{3.22}$$

$$R^{var}_{ack}(m,n) = \sum_{r \in Recv(m)} \frac{G_{mr} d_{mr}}{\sum_{r'} G_{mr'} d_{mr'}} \cdot (1 - L_{mr}) \cdot R^{var}_{rn} \tag{3.23}$$

where L_{mr} is the packet loss rate obtained during the previous iteration and $G_{mr} = \sum_{k=0}^{RMAX} L^k_{mr}$.

Finally, once all the loss rates L_{mn} are available and t_{mn} has been computed, we can compute the goodput as $g_{mn} = \eta t_{mn} \frac{1 - L^{RMAX+1}_{mn}}{G_{mn}}$, where G_{mn} is the average number of transmissions per data packet, $(1 - L^{RMAX+1}_{mn})$ gives the packet delivery rate (after the initial transmission and $RMAX$ retransmissions), and η is used to exclude the overhead due to packet headers and the preamble.

3.7 Obtaining Model Inputs

In this section, we describe how we obtain the various inputs to our model. To estimate pairwise RSS and the external interference at each node, namely R_{sr} and B_r, we measure RSSI at r when only s is transmitting. We only require $O(N)$ measurements because wireless is broadcast medium and all receivers can measure RSSI when a node transmits. From Reis *et al.* [88], $RSSI_{sr} = 10 log_{10}(\frac{R_{sr} + B_r}{W_r})$. For simplicity, we assume $B_r = 0$. (This would be true when interference from external transmitters is negligible.) R_{sr} is then estimated by finding a log-normal

distribution that best fits RSSI measurement data. Let $\bar{R_{sr}}$ and $R_{sr}^{\prime var}$ denote the mean and variance of the best fitting log-normal distribution. The final RSS distribution is estimated as a log normal distribution with mean of $\bar{R_{sr}}$ and variance of $R_{sr}^{\prime var} \cdot \frac{t_{preamble}}{t_{slot}}$. We estimate RSS variance as $R_{sr}^{\prime var} \cdot \frac{t_{preamble}}{t_{slot}}$ because we are interested in RSS variation in the time scale of slots while RSSI is measured as an average over the preamble period and $R_{sr}^{\prime var}$ is $\frac{t_{slot}}{t_{preamble}}$ of the slot-level RSS variance.

As mentioned in Section 3.5.2.1, when the RSS distribution is available, we can estimate $\Pr\{R_{mn} > \gamma_n\}$ immediately from the distribution. In practice, because RSSI measurements are only available on received packets, estimating the true RSS distribution is hard. To get around the problem, we can estimate $\Pr\{R_{mn} > \gamma_n\}$ by directly computing the delivery rate (*i.e.*, the fraction of packets that are received) using the RSSI measurement data.

We find that when the delivery rate is too low (*e.g.*, below 10%), computing the mean and variance of RSS based on RSSI measurements yields significant error because RSSI measurements are only available on received packets. Accurately estimating the true R_{sr} under such cases is an interesting subject on its own, and we leave it as part of our future work. In our current testbed evaluation, we consider only the sender groups such that every node pair s and r within the sender group has either $L_{sr} \leq 90\%$ or $L_{sr} = 100\%$. In the former case, average RSSI is used for estimating R_{sr}, while in the latter case we assume $R_{sr} = 0$. For fair comparison with the UW model, in all 2-sender evaluation we do not apply the above filtering, and compare the estimated and actual values over all sender groups.

Our model also requires the values of a few radio-dependant constants. For testbed experiments, based on our hardware, we use -95 dBm as thermal noise, 2.5 dB as SINR threshold, and -85 dBm as CCA threshold. For simulation experiments, we use the default values in Qualnet, where the thermal noise is -92.52 dBm in 802.11a and -102.5191 dBm in 802.11b, SINR threshold is 2.5 dB, and CCA threshold is -85 dBm in 802.11a and -93 dBm in 802.11b.

3.8 Simulator-based Evaluation

We evaluate the accuracy of our model in both simulation and testbed settings. These two evaluation methodologies are complementary. Testbed experiments allow us to quantify accuracy in more realistic scenarios which are subject to fluctuation in the RF environment, measurement errors, and variations across real hardware. Simulation offers a more controlled environment and allows us to more comprehensively assess the accuracy of individual components in our model. Many of the simplifying assumptions in our model relate to the interaction of the MAC protocol, and any inaccuracy due to these assumptions impact the simulator results as well.

3.8.1 Qualnet Modifications

We use Qualnet 3.9.5 for our evaluation. It has been shown to provide a relatively accurate and realistic simulation environment [99]. We make the following modifications to the Qualnet.

Correct desynchronization problem: The IEEE 802.11 standard states

that when the medium is busy at any time during a backoff slot, the backoff procedure must be suspended without decreasing the value of the backoff timer. However in Qualnet, the backoff timer is decremented by propagation delay and causes time desynchronization. Such desynchronization results in an unrealistically low collision ratio, as reported in [14] and confirmed by our evaluation. We fix the problem by ensuring that the backoff timer is not decremented when the medium is busy at any instant during a time slot.

Disable EIFS: According to the IEEE 802.11 standard, in DCF a frame transmission must use EIFS whenever a frame transmission begins but does not result in the correct reception of a complete MAC frame. However several research papers [14,66] report that EIFS results in unfairness, and suggests disabling EIFS by setting EIFS duration to the same value as DIFS. Existing chipsets such as Atheros also have a configurable EIFS duration. We use the above method to disable EIFS, and postpone modeling EIFS to our future work.

Modify capture effects: In Qualnet, a receiver accepts frames with stronger signals only when they arrive earlier than reception of other frames. Recently, Kochut *et al.* [51] report that real wireless cards accept frames with stronger signals even if they arrive after reception has started. Therefore, we modify Qualnet to accept frames with stronger signals regardless of whether they arrive earlier or later than reception of other frames. In contrast to modifications used in [51], we also accept frames that arrive after preamble of the frame being received. This simplifies our model, and we plan to explore a detailed model of capture effects in our future work.

Support SINR model: The 802.11 implementation in Qualnet uses a Bit-Error-Rate (BER) model, where it first computes SINR of the current packet and uses its SINR to determine BER and convert it to the packet loss rate. In order to match Qualnet simulation, our model needs the same BER table implemented in Qualnet. However Qualnet source code does not reveal the BER table it uses for 802.11. To ensure consistency across our model and Qualnet, we implement the commonly used SINR model in both Qualnet and our model. If the BER table becomes available, our model can immediately support BER model by using BER table to map from SINR to loss rate.

3.8.2 Evaluation Methodology

We evaluate our model for both broadcast and unicast by varying the number of simultaneous senders, the frequency band, and the network topologies. We consider both saturated demands and unsaturated demands. The demand is normalized by the MAC-layer data rate, and a sender with saturated demand has demand of 1.

Throughout the evaluation, we use 25 node topologies. Senders generate 1024-byte UDP packets at a constant bit rate (CBR). The actual sending rate to the air may not be constant, however, due to variable contention delay. We use the lowest MAC data rates, *i.e.*, 6Mbps in 802.11a and 1Mbps in 802.11b. The communication ranges of 802.11a and 802.11b with the lowest data rates are 169 m and 348 m, respectively.

For each scenario, we conduct 10 random runs, where each run randomly se-

lects the senders and receivers and the demands. We use normalized (based on maximum possible) throughput to refer to the total sending rate (including the packet header and preamble) and normalized goodput to refer to the receiving rate (excluding packet overhead). We quantify the accuracy of our model by comparing with the actual throughput and goodput and computing mean absolute error (MAE) and root mean square error (RMSE). MAE is defined as $\frac{\sum_i |est_i - actual_i|}{n}$, and RMSE is defined as $\sqrt{\frac{\sum_i (est_i - actual_i)^2}{n}}$, where n is total number of predictions. We also study the accuracy in detail using scatter plots of actual and estimated values. For clarity, in the scatter plots the data points are plotted in an increasing order of actual values.

We consider the following scenarios below: (i) 2 broadcast senders with saturated demands; (ii) N broadcast senders with saturated demands; (iii) N broadcast senders with unsaturated demands; (iv) N unicast senders with saturated demands; and (v) N unicast senders with unsaturated demands.

For the first scenario, we compare our model with both Qualnet simulation and UW model [88]. The UW model predicts the impact of interference in the presence of two broadcast senders with saturated demands. It is seeded using $O(N)$ measurements similar to ours – each node takes turn to broadcast packets and other nodes log RSSIs and packet delivery rate. Each node obtains its RSSI versus delivery rate profile using these measurements. To predict the impact of two senders trying to send simultaneously, it first estimates the probability with which senders defers based on the RSSIs they receive from each other. To estimate a receiver's goodput from a sender, it uses the standard SINR model, while treating the activity from the second sender as additional interference at the receiver. Since there are no

existing models for the other scenarios, we compare our model only with the actual values obtained in Qualnet.

3.8.3 Broadcast Traffic

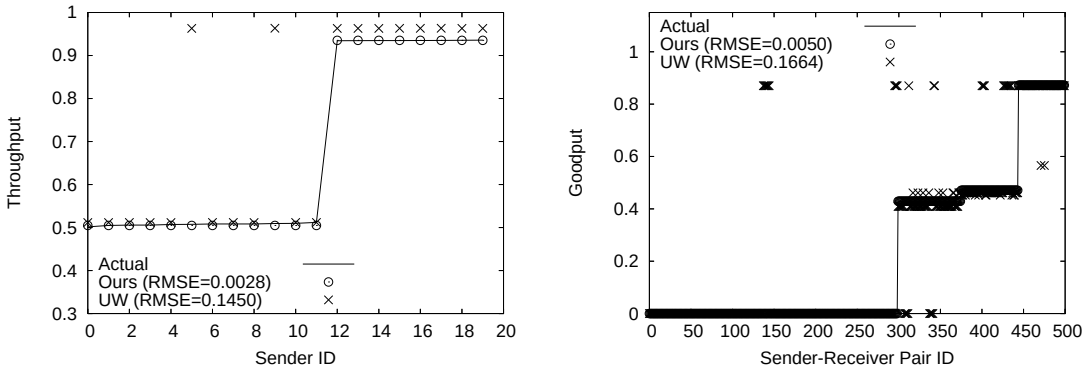
We begin our evaluation by studying broadcast traffic, starting with the simple case of two senders with saturated demands.

3.8.3.1 Two Saturated Senders

Figure 3.1 shows the accuracy of throughput and goodput estimates of our model and the UW model. The graphs plot the actual values obtained in Qualnet and the predictions of the two models. The legend contains the RMSE values for the two models.

We see that both models perform well overall, though our model is more accurate. The RMSE in our model is under 0.5% while that of the UW model is 14% or more. The UW model also tends to have highly inaccurate predictions for a few cases.

The error in UW model is mainly because it assumes a packet can be received as long as its SINR exceeds the threshold. It ignores the other condition that RSS should also exceed the radio sensitivity for a packet to be received. For example, when there is only thermal noise (-95 dBm) and RSS is above -92.5, SINR would be above the 2.5 dB threshold, and the packets are considered to be received 100% of the time under the UW model. However, in reality, when RSS is between -92.5 dBm and -85 dBm (the radio sensitivity value for 802.11a in Qualnet), the



(a) throughput (b) goodput

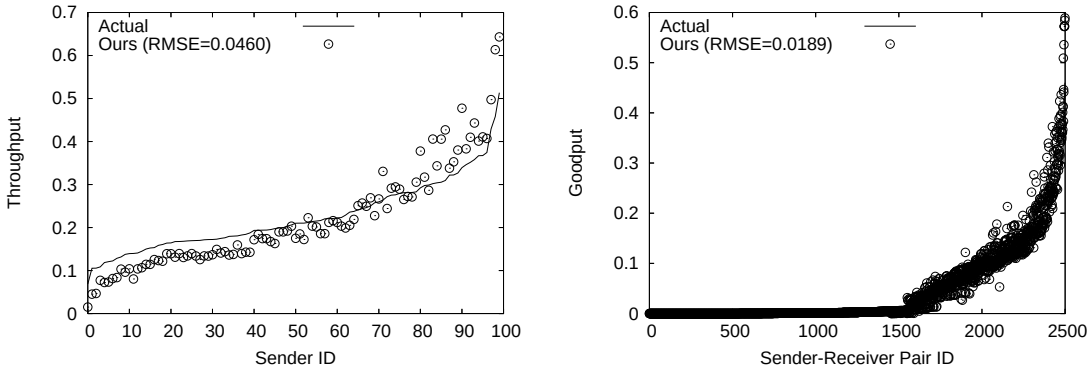
Figure 3.1: 2 saturated broadcast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.

delivery rate is in fact 0. Unfortunately, there is no simple extension to the UW model to accommodate the radio sensitivity constraint because the model builds RF profile directly based on delivery rate. With the radio sensitivity constraint, there is no longer a direct translation between R_{sr} and delivery rate since their relationship changes from $Pr\{\frac{R_{sr}}{I_r+W_r} \geq \delta_r\}$ to $Pr\{R_{sr} > \gamma_r \; and \; \frac{R_{sr}}{I_r+W_r} \geq \delta_r\}$. For a given delivery rate, R_{sr} is no longer unique.

3.8.3.2 N Saturated Senders

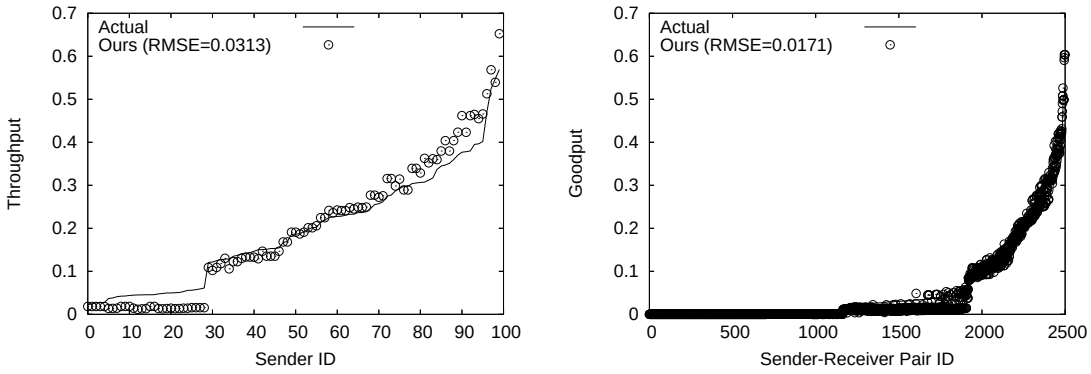
Next, we consider the case of N broadcast senders. Each sender has saturated demand, as before. We evaluate our model by varying the frequency band, network topology, and the number of senders.

Different frequency bands (802.11a and 802.11b): Figures 3.2 and 3.3 show the scatter plots of actual and predicted throughput and goodput under 802.11a and 802.11b. In each case, there are 10 broadcast senders with infinite demands. We



(a) throughput (b) goodput

Figure 3.2: 10 saturated broadcast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.

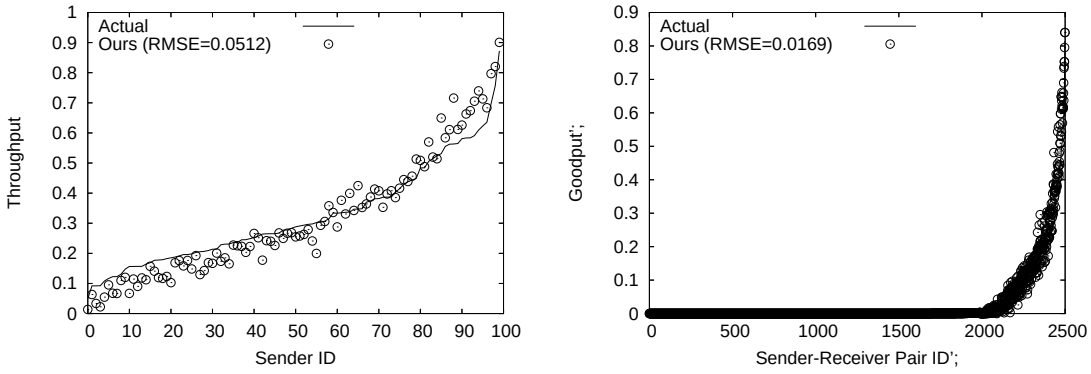


(a) throughput (b) goodput

Figure 3.3: 10 saturated broadcast senders using 802.11b in a 5×5 grid topology in an $500m \times 500m$ area.

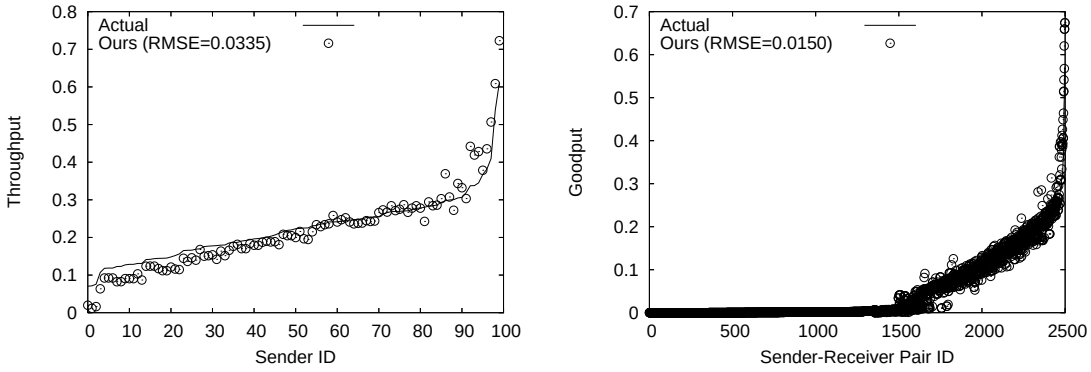
see that our model is highly accurate in both cases, with less than 5% RMSE. The goodput error is lower than the throughput error because many receivers have no connectivity to one or more senders, and it is easier to predict the exact goodput for such receivers.

Different network topologies (grid and random): Figure 3.4 and 3.5 show the



(a) throughput (b) goodput

Figure 3.4: 10 saturated broadcast senders using 802.11a in a 5×5 grid topology in an $500m \times 500m$ area.

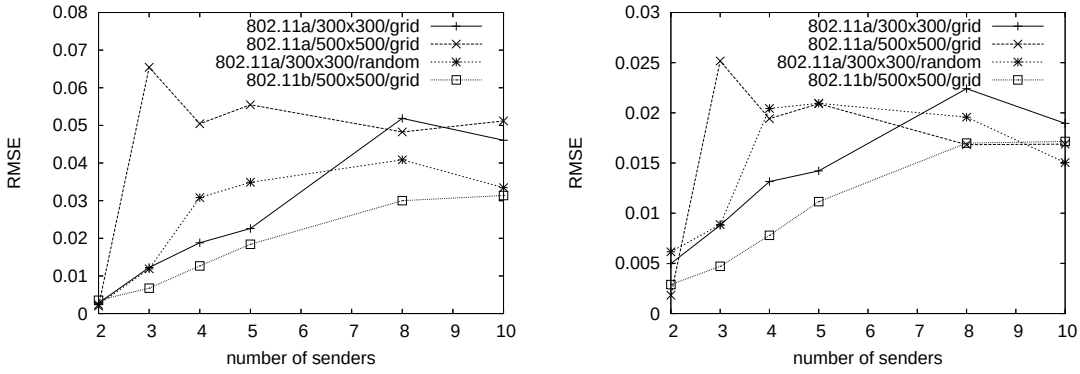


(a) throughput (b) goodput

Figure 3.5: 10 saturated broadcast senders using 802.11a in random topologies, where nodes are randomly placed in an $300m \times 300m$ area.

results for 10 broadcast senders using 802.11a in a $500m \times 500m$ grid topology and $300m \times 300m$ random topology. In each case, the model closely tracks the actual values and the error is around or below 5%.

Different number of senders (2-10): Figure 3.6 plots throughput and goodput RMSE as a function of the number of broadcast senders. We see that the error



(a) throughput (b) goodput

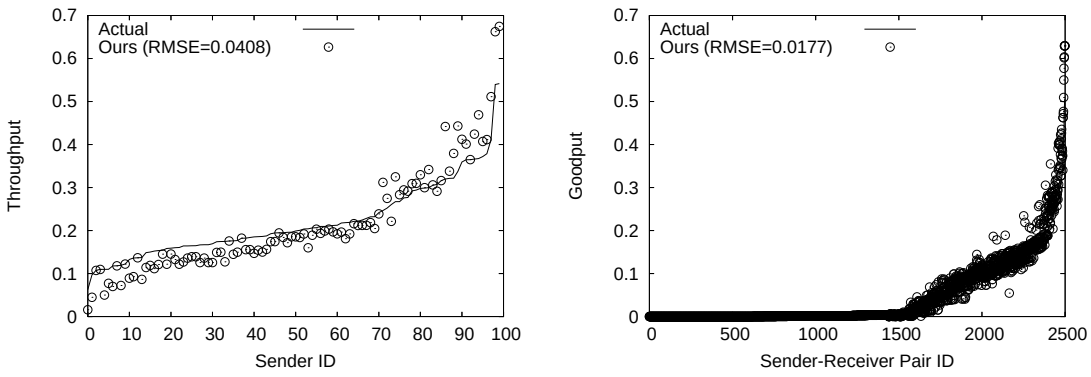
Figure 3.6: RMSE under a varying number of sender.

tends to increase slightly with the number of senders due to more complex interactions. Yet under all numbers of senders, the model can keep RMSE within 7% for throughput estimation and within 3% for goodput estimation.

3.8.3.3 N Unsaturated Senders

We now consider unsaturated senders and allow nodes to have different traffic demands. We assign each sender a normalized demand between 0.1 and 0.9 and use the corresponding inter-arrival time for CBR traffic.

Figure 3.7 shows the results for 10 broadcast senders using 802.11a in a 5×5 grid topology over a $300m \times 300m$ area. We see that the accuracy of our model for unsaturated demands, which are harder to model, is high as well and comparable to its accuracy for saturated demands.



(a) throughput (b) goodput

Figure 3.7: 10 unsaturated broadcast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.

3.8.4 Unicast Traffic

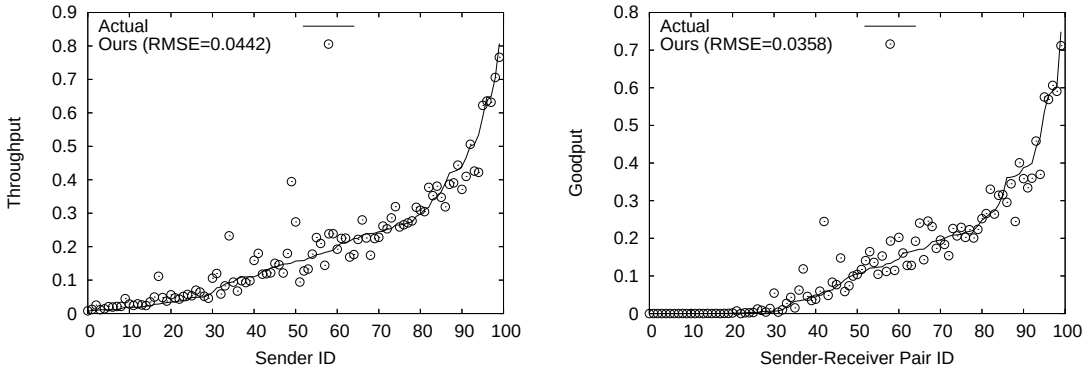
In this section, we turn our attention to unicast traffic and evaluate how well the unicast extensions of our model perform.

3.8.4.1 N Saturated Senders

We start with the case of N saturated senders. Figure 3.8 shows the result for 10 unicast senders using 802.11a. As for broadcast traffic, the predictions of our model track the actual values closely, and the RMSE is within 5%.

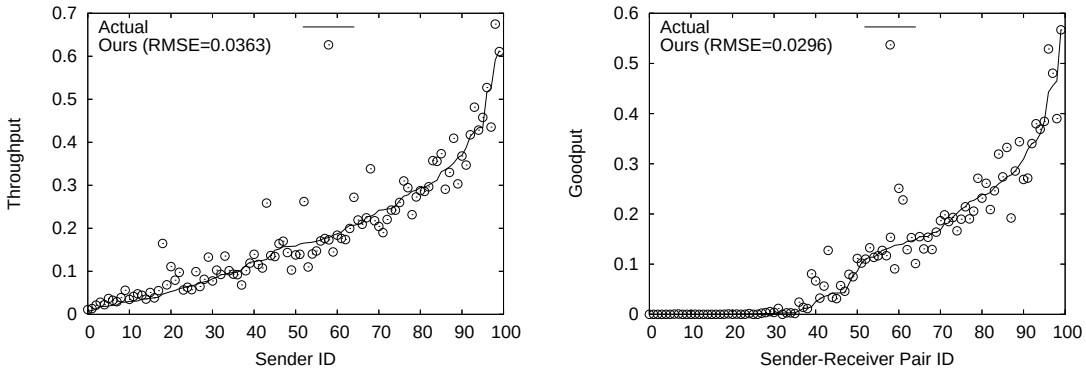
3.8.4.2 N Unsaturated Senders

We conclude our simulation-based evaluation by studying the case of unsaturated unicast senders. As above, we have 10 senders using 802.11a in a 5×5 grid topology. The demand for each sender is assigned as for the broadcast setting in Section 3.8.3.3.



(a) throughput (b) goodput

Figure 3.8: 10 saturated unicast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.



(a) throughput (b) goodput

Figure 3.9: 10 unsaturated unicast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.

Figure 3.9, shows the prediction results for this setting. Our model continues to yield accurate predictions. Not only is the net RMSE under 4%, but we also do not have individual instances where the predictions of our model are highly inaccurate.

Summary In this section, we used simulation to evaluate the accuracy of our model in many diverse settings which include broadcast and unicast traffic, unsaturated and saturated demands, and different number of senders. We find our model's predictions of throughput and goodput are accurate in all the settings that we considered, and its RMSE value is typically under 5%. We also find that our model, while being more general, is also more accurate than a state-of-art model [88] for the specific case of 2 broadcast senders with saturated demands.

3.9 Testbed-based Evaluation

In this section, we evaluate our model using testbed experiments. Our goal is to quantify the accuracy of our model in real RF environments and with real hardware. We employ traces from two different testbeds for this purpose. Below, we describe these testbeds and the traces, followed by the evaluation results for each testbed.

3.9.1 Testbeds and Traces

The two testbeds are our own indoor wireless testbed and the UW testbed used by Reis *et al.* [88]. Our testbed has 22 DELL dimensions 1100 PCs, located on the same floor of an office building. Each machine has a 2.66 GHz Intel Celeron D Processor 330 with 512 MB of memory, and is equipped with 802.11 a/b/g NetGear WAG511. Each machine runs Fedora Core Linux. We use *Madwifi* as the driver for the wireless cards, and use *click* to collect traces.

We collect the trace as follows. First, we let one node broadcast 1000-byte

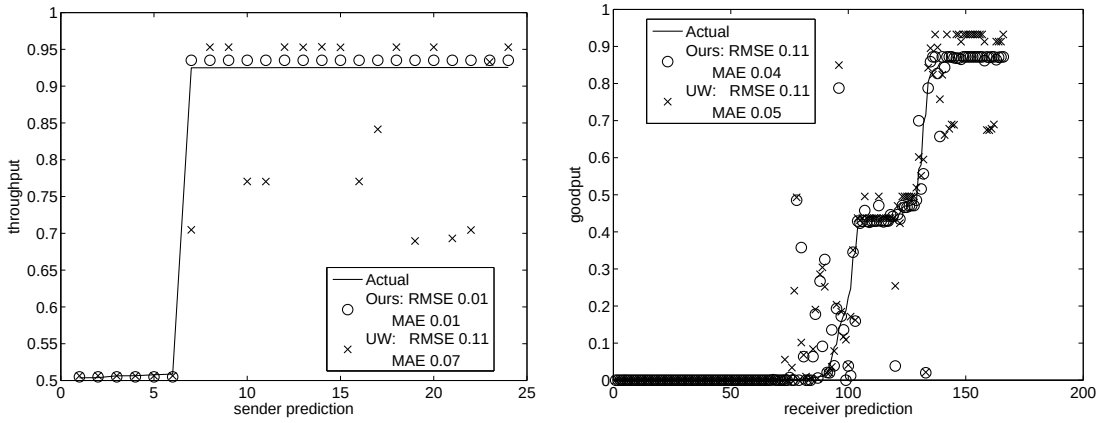
UDP packets at full speed for 1 minute and log received packets at all the other nodes. We repeat the process until every node in the testbed has broadcast once. We refer to this as 1-sender trace. Applying the approach described in Section 3.7 to the 1-sender trace gives us estimate of RSS between every pair of nodes and external interference at each node. Since there is a resident 802.11b/g wireless network that causes strong interference, we collect traces using only 802.11a on our testbed. Unless otherwise specified, each node uses 30 mW transmission power.

In order to evaluate the accuracy of our model, we measure the actual sending and receiving rates under multiple senders. These traces are only needed for obtaining "ground truth" and not required for using the model. Given a specified number of senders k, we randomly select k nodes and let them broadcast simultaneously for 1 minute. All other nodes log received packets. In the 1-minute broadcasting period, the nodes send as fast as possible for the saturated demand experiments. For unsaturated demands, each sender is assigned a normalized demand which is total demand divided by the MAC data rate. The normalized demand is selected randomly between 0.1 and 0.9 and specifies the maximum rate at which the sender can send. For each configuration, *i.e.*, the specified number of senders and demand type, we conduct 100 random runs with different set of k senders.

The UW testbed had 14-nodes inside an office building. The traces we use are same as those used for evaluating the UW model [88]. The collection methodology is similar to the above except that these traces contain only 2 broadcast senders with saturated demands. We study both 802.11a and 802.11b using these traces.

3.9.2 The UW Testbed

We first present the results for the UW testbed in this section and then for our testbed in the next section. Figure 3.10 shows scatterplots of predicted and actual throughput and goodput under 802.11a.

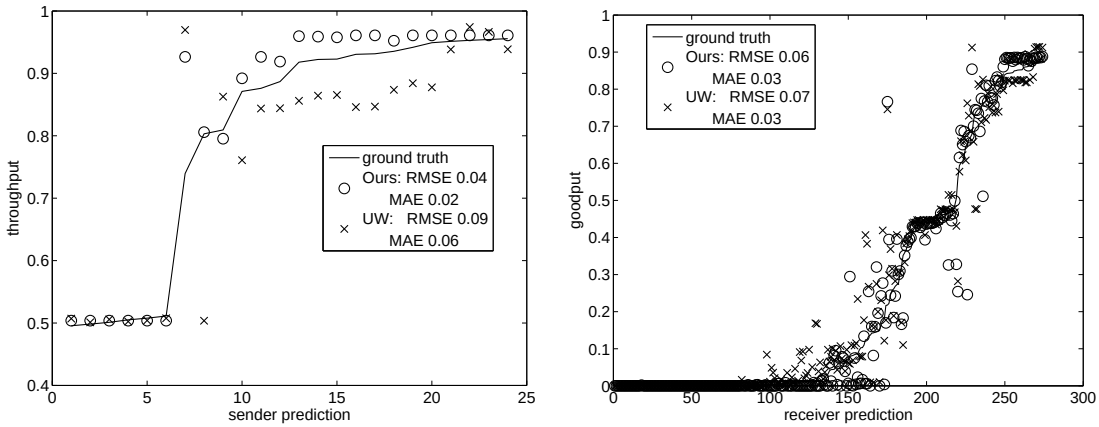


(a) throughput (b) goodput

Figure 3.10: 2 saturated senders using 802.11a in UW traces.

As we can see, our model closely tracks the actual throughput and goodput. UW model has higher error in the throughput prediction. Most mispredictions occur when the UW model incorrectly predicts that two senders defer to each other. This error is caused by the linear interpolation heuristics to estimate delivery probability for a hypothetical RSSI [88]. The heuristic implicitly assumes delivery probability is linearly proportional to RSSI, which may not be true in reality. Interestingly, UW model has comparable accuracy to our model in goodput prediction. A closer look reveals that for many links that have higher throughput error, their goodput is often close to 0 due to poor link quality. Such cases are easy to predict, which reduces

overall goodput error.



(a) throughput (b) goodput

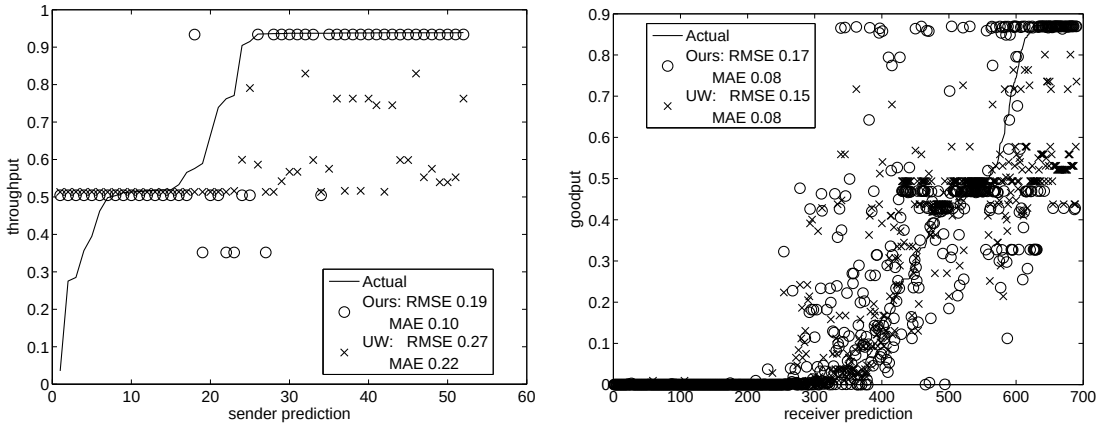
Figure 3.11: 2 saturated senders using 802.11b in UW traces.

Figure 3.11 shows the results for 802.11b. As for 802.11a, our model has more accurate throughput prediction than the UW model, while both models have comparable prediction errors for goodput.

3.9.3 Our Testbed

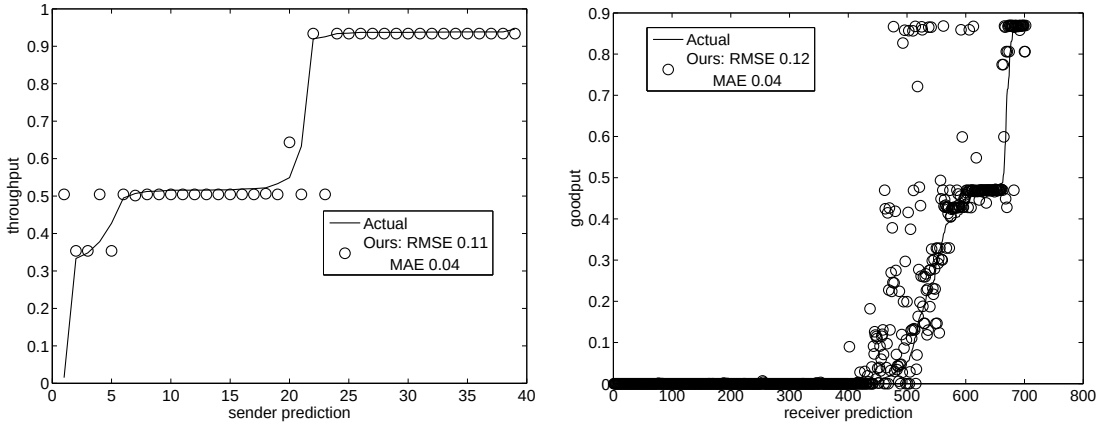
For our testbed, we evaluate our model by varying number of senders and using both saturated and unsaturated demands. Figure 3.12, 3.13, 3.14, and 3.15 show scatter plots of throughput and goodput under 2, 3, 4 and 5 senders with saturated broadcast demands.

Since the UW model is only applicable to 2 senders, we compare with the UW model only for 2 senders. As we can see, our model tracks the actual throughput more closely than the UW model, and yields comparable accuracy for goodput



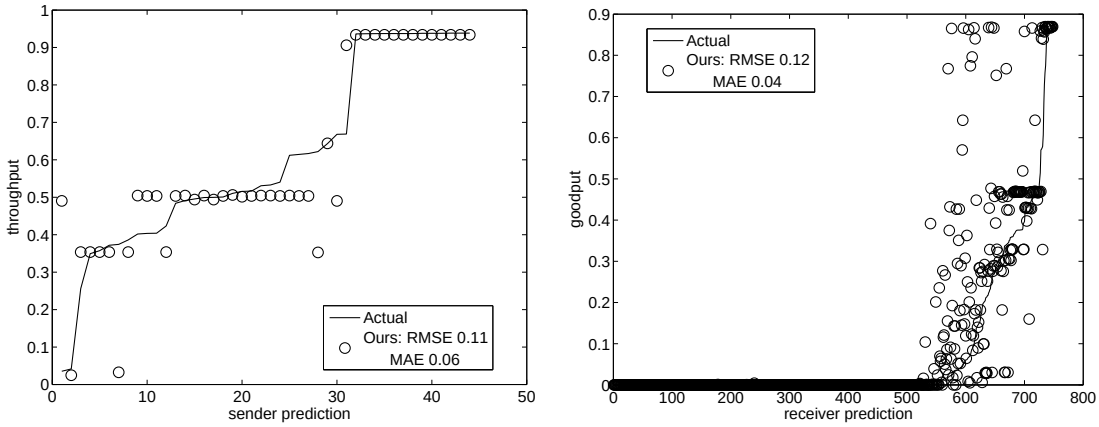
(a) throughput (b) goodput

Figure 3.12: 2 saturated broadcast senders using 802.11a in our traces.



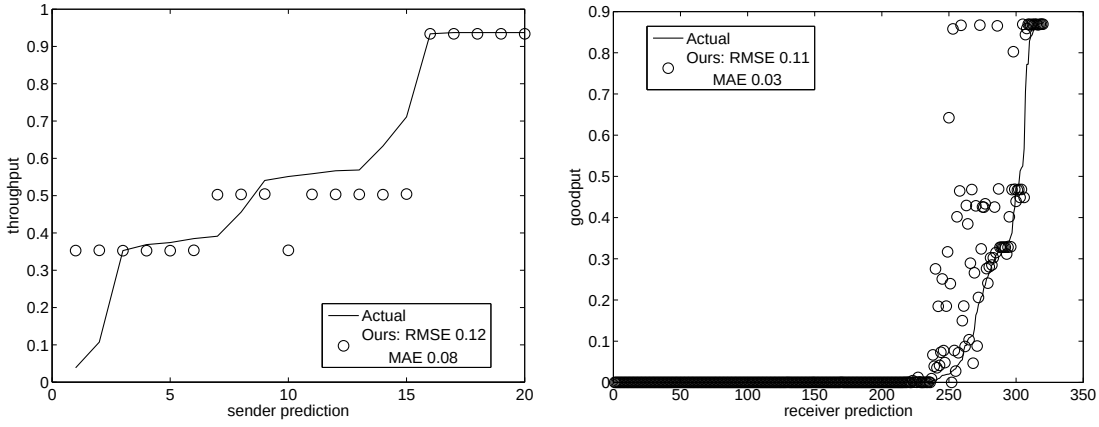
(a) throughput (b) goodput

Figure 3.13: 3 saturated broadcast senders using 802.11a in our traces.



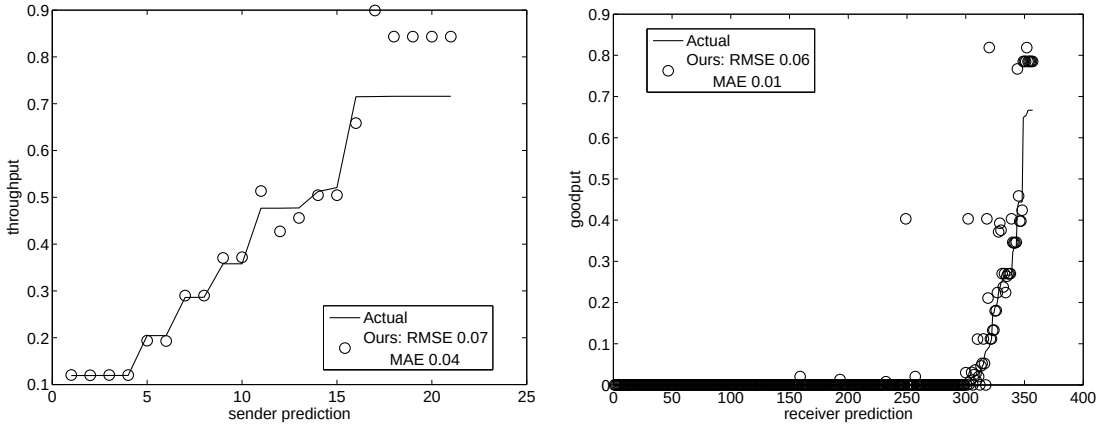
(a) throughput (b) goodput

Figure 3.14: 4 saturated broadcast senders using 802.11a in our traces.



(a) throughput (b) goodput

Figure 3.15: 5 saturated broadcast senders using 802.11a in our traces.



(a) throughput (b) goodput

Figure 3.16: 3 unsaturated broadcast senders using 802.11a in our traces, where each sender uses 1 mW.

prediction. This is also reflected in RMSE and MAE. For 3, 4, and 5-sender cases, our model yields estimation close to the actual rates: its RMSE is within 0.12 and its MAE is within 0.06.

Figure 3.16 shows the results for unsaturated demands, with 3 senders. As for saturated demands, our model maintains high accuracy: its RMSE is within 0.07 and MAE within 0.04.

Summary The testbed evaluation confirms that our model works well in real environments and using real hardware. Compared with simulation, predicting testbed performance is much more challenging due to factors such as biased and noisy measurements, as well as environmental variation. Despite these challenges, the results show that our model is effective in predicting throughput and goodput.

Chapter 4

Small State and Small Stretch Routing

4.1 Overview

Routing finds paths in a network along which to send data. It is one of the basic network functionalities. The effectiveness of routing protocols directly affects network scalability, efficiency, and reliability. With continuing growth of wireless network sizes, it is increasingly important to develop routing protocols that *simultaneously* achieve the following design goals.

- Small routing state: Using small amounts of routing state is essential to achieving network scalability. Many wireless devices are resource constrained. For example, mica2 sensor motes have only 4KB RAM. Limiting routing state is necessary for such devices to form large networks. Moreover, limiting routing state also helps to reduce control traffic used in route setup and maintenance, since the amount of routing state and control traffic is often correlated.

- Small routing stretch: Routing stretch is defined as the ratio between the cost of selected route and the cost of optimal route. Small routing stretch means that the selected route is efficient compared to the optimal route. It

is a key quantitative measure of route *quality*, and affects global resource consumption, delay, and reliability.

- Resilience: Wireless networks often experience frequent topology changes arising from battery outage, node failures, and environmental changes. Routing protocols should find efficient routes even in the presence of such changes.

Existing routing protocols either achieve small worst-case routing stretches with large routing state (*e.g.* shortest path routing) or achieve small routing state at the cost of large worst-case routing stretches (*e.g.* geographic routing and hierarchical routing). In this chapter, we present the design and implementation of Small State and Small Stretch (S4), a new addition to the routing protocol design space. S4 achieves a desirable balance among these characteristics, and is well suited to the wireless sensor network setting.

We make the following contributions.

1. S4 is the first routing protocol that achieves a worst-case routing stretch of 3 in large wireless networks. Its average routing stretch is close to 1.

2. S4's distance guided local failure recovery scheme significantly enhances network resilience, and is portable to other settings.

3. S4's scalability, effectiveness of resource use, and resilience are validated using multiple simulation environments and a 42-node sensor network testbed.

4.2 Related Work

Routing is a well-studied problem, but large-scale wireless networks have introduced new challenges. Traditional shortest path routing protocols based on distance vector or link state algorithms are effective for small networks, but scale poorly to large networks due to both overwhelming control traffic and the amount of state to keep at each node. To reduce the overhead, reactive on-demand routing protocols have been proposed, such as DSR [44] and AODV [80]. Their overhead depends on traffic demands, and they do not work well when there are many source-destination pairs. As shown in [20], DSR and AODV generate more control traffic than data traffic in 100 nodes with 40 source-destination pairs. Consequently, routing in large-scale wireless networks has focused on minimizing storage and exchange of routing state. In this section, we briefly review the literature of scalable routing in these categories: (i) geographic routing, (ii) hierarchical routing, (iii) DHT-type routing, and (iv) theoretical work on scalable routing.

Geographic routing: In geographic routing, each node is assigned a coordinate reflecting its position in the network. Upon receiving a packet, a node selects a next hop closer to the destination than itself in the coordinate space. Some geographic routing protocols use geographic locations as node coordinates, while others use virtual coordinates based on network proximity. These schemes must address the problem of getting "stuck" in a local minimum, where no neighbor is closer to the destination than the current node. Some proposals such as GFG [10], GPSR [45], GOAFR+ [56], GPVFR [61] and variants use face traversal schemes

that route packets on a planar graph derived from the original connectivity graph. Their delivery guarantees [27] depend on the assumption that the planarization algorithms (*e.g.* GG [28] and RNG [102]) can successfully planarize *any* network graph. These planarization algorithms typically assume a unit disk or quasi-unit disk model. However, these models can be inadequate for real wireless environments due to obstacles and multi-path fading. Kim *et al.* [49] have shown that model failures in real radio environments can cause routing pathologies and persistent routing failures. CLDP [48] addresses the imperfect RF propagation problem using a right-hand probing rule to detect link-crossings and remove them to re-planarize the graph. The correctness of CLDP comes at a cost of probing each link multiple times.

GDSTR [62] provides delivery guarantee without requiring planarization by avoiding routing across the face of planar graphs. Instead packets are routed through a spanning tree. Each node of the tree is annotated with a convex hull as the location aggregation of its subtree. The convex hulls are used to determine routing directions when routing over the spanning tree. The effectiveness of GDSTR has yet to be demonstrated in real networks. One of the major concerns is that, due to irregular communication range, the hulls of many pairs of siblings nodes in a tree may intersect and therefore significantly degrade the efficiency.

The geographic coordinate-based routing schemes have at least three difficulties for wireless sensor networks. First, accurate geolocation either requires careful static setting or access to GPS, with consequences for cost or need for line-of-sight to satellites. Second, geographic distances may lack predictive value for

network performance (*e.g.* loss rate). This may result in paths with poor performance. Third, even with GPS and ideal radios, the best routing stretch for geographic routing is $O(c)$ in GOAFR+ [56] and ARF [57], where c is the length of the optimal path. Example topologies exist where this bound is tight [57].

Virtual coordinates reflecting underlying network connectivity address the first two difficulties, but still face the challenge of "dead ends", for which a recovery scheme is required. In addition, the overhead of computing and storing virtual coordinates is not negligible. For example, NoGeo [86] uses $O(\sqrt{N})$ *perimeter* nodes to flood the N-node network so that every node can learn its distances to all the perimeter nodes. Each node determines its virtual coordinate based on the distances to the perimeter nodes. However, perimeter nodes need to store $O(N)$ pair-wise distances among them. It is not scalable in large wireless networks with limited memory space per node. GEM [72] achieves greater scalability by using triangulation from a root node and two other reference nodes. However, the routing stretch is larger than that typical of geographic routing algorithms, and there is the additional cost of recomputing routing labels resulting from network failures.

Fonseca *et al.* [26] have proposed Beacon Vector Routing (BVR) which selects a few beacon nodes, and uses flooding to construct spanning trees from the beacons to all other nodes. A node's coordinate is a vector of distances to all beacons, and each node maintains the coordinates of its neighbors. BVR defines a distance metric over these beacon vectors, and a node routes packets to the one that minimizes the distance. When greedy routing stalls, it forwards the packet towards the beacon closest to the destination. If the beacon still fails to make greedy

progress, scoped flooding is used. One of the drawbacks of BVR is that each packet is annotated with a full-length beacon vector (*e.g.*about 40 entries in a 3200-node network as suggested in the paper), which is significant overhead.

None of the virtual coordinate-based routing algorithms provide worst-case routing stretch guarantees. Furthermore, virtual coordinates change with wireless network conditions, which may incur significant control overhead.

Hierarchical routing: Hierarchical routing is an alternative approach to achieving scalability. Nodes in a network are divided into clusters. There may be two or more levels of hierarchies. Typically, each node maintains full topological information about its local cluster, but only maintains little topological information about nodes in other clusters. Therefore, routing inside a cluster is optimal, but routing towards other clusters may traverse a sub-optimal path. Many existing hierarchical routing protocols have been proposed, including landmark routing [103], HSR [76], LANMAR [32], ARCH [7], Safari [82] and ZRP [38].

Landmark routing is based on Landmark Hierarchy, which can be dynamically configured. There are a subset of nodes, called landmarks, in the network. A landmark maintains routing state to other landmarks within certain radius. Initially, each router is a landmark of level 0. A subset of level 0 landmarks are landmarks of level 1, and so on. Higher level of landmarks have larger radius. The radius of landmarks at highest level is at least the diameter of the network. These are called global landmarks since all routers can see them. Each node maintains routing state for landmarks at different levels within corresponding radius. When routing to-

wards a destination, a node looks up its routing table and routes toward the lowest level landmark common with the destination. HSR, LANMAR, ARCH and Safari exploit similar idea of hierarchical clustering. None of them provide routing stretch guarantee due to the boundary effect: two nodes that are physically close may belong to different clusters, hence the route between them has to go through cluster heads or landmarks and can be arbitrarily longer than the shortest path.

The clustering technique of ZRP is different. In ZRP, each node maintains an individual cluster which contains all nodes within a zone radius. Implicitly, there are just two levels of hierarchy. Each node proactively maintains routing state for all nodes in its own cluster. For destinations outside the radius, ZRP uses a query-reply scheme to establish routes as in reactive on-demand routing protocols. Although ZRP can achieve efficient routing stretch for nodes far apart, it may incur large control traffic overhead as the network scale increases due to on demand routing request. Furthermore, it is not a trivial decision to make whether to store hard routing state for destinations outside local clusters. If stored, the routing state may get arbitrarily large. If not stored, the control traffic overhead may recur for every destination every time. Overall, it is difficult for ZRP to achieve a desired tradeoff between routing state and control overhead.

DHT-type routing: Caesar *et al.* develop VRR [12], a network layer point-to-point routing protocol inspired by distributed hash tables. Each node is assigned a location-independent identifier. All nodes are organized into a virtual ring in increasing order of their identifiers. Each node maintains a virtual neighbor set

containing half closest neighbors clockwise and half counter clockwise. Each node sets up and maintains routing paths to all virtual neighbors. In addition, each node also maintains state about paths traversing through it. In the routing table, each entry specifies two endpoints of a path and the next hop towards each endpoint. The basic routing strategy of VRR is similar to greedy forwarding in geographic routing protocols. Among all endpoints in the routing table, the one with identifier closest to destination is chosen and packets are forwarded to the next hop towards this endpoint. Since each node maintains state to virtual neighbors on both sides on the ring, there always exists such an endpoint until reaching the destination. This forwarding scheme is also used to initialize the paths to virtual neighbors when each node joins the networks. Therefore, VRR does not require any flooding in the network. However, VRR still does not provide worst-case routing stretch guarantee, since the proximity on the virtual ring is independent of the proximity in the physical network.

WSR [1] is another DHT-type routing protocol designed for large scale highly dynamic networks. It requires location information of all nodes. It aggregates information about a set of remote locations in a same region, by mapping a set of ID to a region. The ID-to-region mappings are represented by weak Bloom Filters. The routing task is accomplished using unstructured random directional walks. Intermediate nodes prioritize their ID-to-region mappings to bias and forward the random walk. WSR can tolerate dynamic changes of the network because the weak state does not require hard limit on expiration. However, WSR may increase path length compared to traditional geographical routing protocols.

Theoretical work on scalable routing: Theoretical work [19, 101] on achieving scalable and efficient routing has developed *compact routing* algorithms that provide a worst-case routing stretch of 3 while using at most $O(\sqrt{N \log N})$ state in an N-node network. This worst-case routing stretch is provably optimal when each node uses less than linear routing state [19,101]. While compact routing seems to be a promising direction for large-scale networks, it cannot be directly translated into a routing protocol in a distributed network. In particular, the proposed algorithms do not specify how each node should build and maintain routing state for local clusters and for beacon nodes. Moreover, the algorithm in [101] requires choosing beacon nodes offline, considers only initial route construction, and cannot cope with topology changes, which precludes realization in our network setting. The implications of compact routing for average routing stretch also remain unclear.

4.3 S4 Routing Protocol

S4 uses the theoretical ideas of the compact routing algorithm [101] as a basis, refined by the addition of new techniques needed to obtain a practical routing protocol for large-scale wireless networks. We first describe the basic routing algorithm and note challenges for routing protocol design, and then present the S4 routing protocol. Throughout this chapter, our metric for the cost of a route is the number of links traversed (*i.e.* hop count).

4.3.1 Basic Routing Algorithm

In S4, a random set of nodes, L, are chosen as beacons. For a node d, let $L(d)$ denote the beacon closest to node d, and let $\delta(s, d)$ denote the shortest path distance from s to d. Each node s constructs the following local cluster, denoted as $C_k(s)$.

$$C_k(s) = \{c \in V | \delta(c, s) \leq k * \delta(c, L(c))\}, k \geq 1.$$

where V is the set of all nodes in the network. A local cluster of node s consists of all nodes whose distances to s are within k times their distances to their closest beacons. Each node s then maintains a routing table for all beacon nodes and nodes in its own cluster $C_k(s)$.

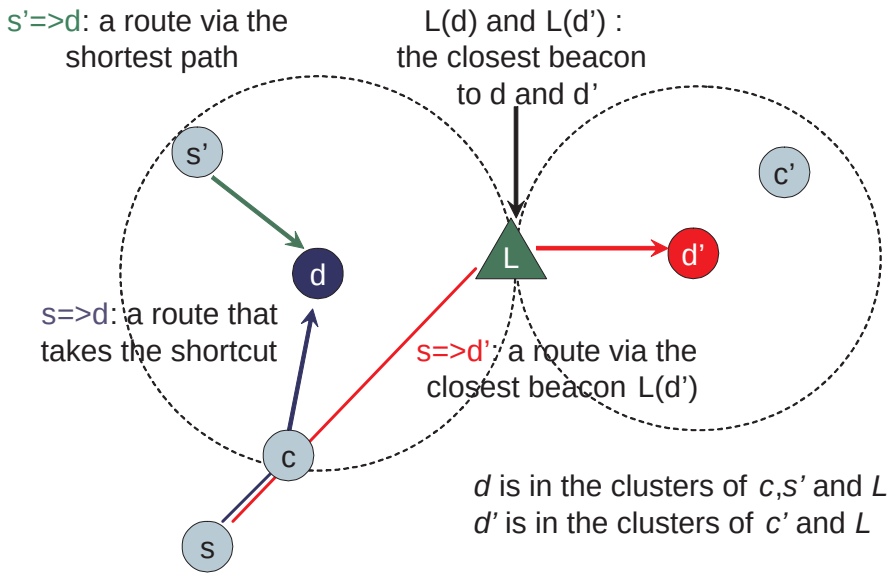


Figure 4.1: S4 routing examples. Every node within the circle of d has d in its local cluster. The route $s' \rightarrow d$ is the shortest path; the route $s \rightarrow d$ takes a shortcut at c before reaching $L(d)$; the route $s \rightarrow d'$ is through $L(d')$ without shortcut.

As shown in Figure 4.1, when routing from node s to node d, if $d \in C_k(s)$, we can directly use the shortest path to route from s to d. Otherwise, s first takes the shortest path towards $L(d)$, and then use the shortest path to route towards d. In the second case, the route does not have to always reach $L(d)$ before routing to d. Whenever data reaches a node c whose cluster contains d, c can directly route to d using the shortest path from c to d. According to the triangle inequality, the "shortcut" strictly improves routing stretch. We give the following theorem as an extension to the proof in [19, 101], in which a special case $k = 1$ is proved.

Theorem 1. *Let* $C_k(s) = \{c \in V | \delta(c, s) < k * \delta(c, L(c))\}$*, where* $k \geq 1$*. If each node* s *maintains next-hop for the shortest path to every beacon and every node in* $C_k(s)$*, the worst-case routing stretch is* $1 + \frac{2}{k}$*.*

Proof. When $d \in C_k(s)$, routing stretch is 1, since we know the shortest path from s to d. When $d \notin C_k(s)$, let $r(s, d)$ denote the cost of selected route from s to d.

$$
\begin{aligned}
r(s,d) &\leq \delta(s, L(d)) + \delta(L(d), d) && (4.1)\\
&\leq \delta(s,d) + 2\delta(L(d), d) && (4.2)\\
&\leq \delta(s,d) + \frac{2}{k}\delta(s,d) && (4.3)\\
&= (1 + \frac{2}{k})\delta(s,d) && (4.4)
\end{aligned}
$$

The first inequality is due to possible shortcut before reaching $L(d)$. As shown in Figure 4.1, the shortcut $c \rightarrow d$ is less than $c \rightarrow L(d) \rightarrow d$ according to triangle inequality. Hence $s \rightarrow c \rightarrow d$ is less than $s \rightarrow L(d) \rightarrow d$. Equality holds when there is no shortcut. The second inequality is due to triangle inequality and

symmetry: the shortest path $s \rightarrow L(d)$ should cost no more than $s \rightarrow d \rightarrow L(d)$. Finally the third inequality is based on the definition of cluster $C_k(s)$ and the fact that $d \notin C_k(s)$. This completes the proof. □

As a special case, when $k = 1$, a local cluster of node s consists of all nodes whose distances to s are closer than their distances to their closest beacons. This special case is called compact routing [19, 101]. It is particularly interesting, since it has low worst-case storage cost of $O(\sqrt{N \log N})$ and provides a worst-case routing stretch of 3. In the remaining chapter we consider $k = 1$, since it gives small routing state.

Practical concerns dictate three changes to the TZ compact routing scheme [101] to achieve S4. First, the boundary conditions of the cluster definitions are slightly different. In S4, $C(s) = \{c \in V | \delta(c, s) \leq \delta(c, L(c))\}$, but in the TZ scheme, $C(s) = \{c \in V | \delta(c, s) < \delta(c, L(c))\}$. That is, node c is in the cluster of s in S4 but not in the TZ scheme, if $\delta(c, s) = \delta(c, L(c))$. This change does not affect the worst-case routing stretch, and reduces average-case routing stretch at the cost of increasing routing state.

Second, to route towards node d, only $L(d)$ should be carried in the packet header as the location information in S4. In comparison, the TZ scheme requires a $label(d) = (L(d), port(L(d), d))$ for each packet, where $port(L(d), d)$ is the next hop at $L(d)$ towards d. Only with the label carried in the packet header, a beacon node can forward a packet towards d using next hop $port(L(d), d)$. It is necessary in the TZ scheme because the beacon nodes do not store routing state. However, in

S4, as a result of the boundary condition change, each beacon node L stores routing state to all the nodes that have L as its closest beacon node. Given that the total storage cost of the additional field $port(L(d), d)$ in the labels is the same as the total number of routing entries at beacon nodes in S4 (*i.e.* both are N), we favor storing routing state at beacon nodes since it reduces packet header length and the frequency of updating labels. The frequency of label updates is reduced because labels are updated only when $L(d)$ changes but not when $port(L(d), d)$ changes.

Finally, the TZ scheme proposes a centralized beacon node selection algorithm to meet expected worst case storage bound $O(\sqrt{NlogN})$ in an N-node network. Since practicality is our main design goal, in S4 we randomly select beacon nodes in a distributed fashion. It is proved that when $O(\sqrt{N})$ nodes are randomly selected as beacon nodes, the average storage cost on each node is still $O(\sqrt{N})$ [100]. As our evaluation results show, the storage cost is still low even for the worst cases. Note that the worst-case routing stretch of 3 still holds under random beacon node selection.

4.3.2 Design Challenges

Designing a routing protocol to realize the algorithm proposed in Section 4.3.1 poses the following challenges:

First, how to construct and maintain routing state for a local cluster? Unlike traditional hierarchical routing, each node has its own cluster in compact routing. Therefore naive routing maintenance could incur significant overhead.

Second, how to construct and maintain routing state for beacon nodes?

Knowledge of next-hops and shortest path distances to beacon nodes is important to the performance of S4. When beacon packets are lost, the routing state could be inaccurate, which could substantially degrade the performance.

Third, how to provide resilience against node/link failures and environmental changes? Maintaining up-to-date routing state could be expensive especially in a large network. Moreover routing changes take time to propagate. During the transient period (*e.g.*, the period from the time when failure occurs to the time when the routing tables at all nodes are updated to account for the failure), many packets could be lost without a failure recovery scheme.

To address the above challenges, S4 consists of the following three major components: (i) scoped distance vector for building and maintaining routing state to nodes within a cluster, (ii) resilient beacon distance vector for efficient routing towards beacon nodes and facilitating inter-cluster routing, and (iii) distance guided local failure recovery for providing high quality routes even under dynamic topology changes. Below we will describe these three components in turn.

4.3.3 Intra-Cluster Routing: Scoped Distance Vector (SDV)

In S4, node s uses the shortest paths to route towards nodes in the cluster of s. Unlike the traditional hierarchical routing, in S4 each node s has its own cluster, which consists of nodes close to node s. This clustering is essential for providing a routing stretch guarantee, since it avoids boundary effects. In comparison, hierarchical routing cannot provide routing stretch guarantee due to boundary effects, where two nearby nodes belong to different clusters and the hierarchical route be-

tween them could be much longer than their direct shortest path.

A natural approach to building a local routing table is to use scoped flooding. That is, each node d floods the network up to $\delta(d, L(d))$ hops away from d, where $\delta(d, L(d))$ is the distance between d and its closest beacon $L(d)$. Scoped flooding works fine when the network is initialized, or when there are new nodes joining the network. But it is costly to send frequent scoped flooding to reflect constant topology changes, which often arises in wireless networks due to battery outage, node failures, and environmental changes.

Scoped distance vector: To provide cheap incremental routing updates, we propose using scoped distance vector (SDV) for constructing routing tables for local clusters. SDV is attractive because it is fully distributed, asynchronous, and supports incremental routing updates. SDV is more efficient than scoped flooding especially under small changes in a network topology, because a node in SDV propagates routing update only when its distance vector changes while in scoped flooding a node propagates a flooded packet regardless of whether its distance and next hop to a destination have changed.

In S4, each node s stores a distance vector for each destination d in its cluster as the following tuple:

$$< d, nexthop(s, d), \delta(s, d), seqno(d), scope(d), updated >$$

where d and $nexthop(s, d)$ are both node ID, $seqno$ is the latest sequence number for destination d, and $scope(d)$ is the distance between d and d's closest beacon,

and $updated$ is whether the distance vector has been updated since the last routing update.

A node s exchanges its distance vectors with its neighbors either synchronously or asynchronously. Node s initializes $\delta(s, c) = 1$ for only $c \in neighbor(s)$, and ∞ otherwise. Upon receiving a distance vector, a node c uses the newly received distance vectors to update its routing state. Node c further propagates the update for s only when its current distance from s is below $scope(s)$ and its distance vector to s has changed.

Benefits of SDV: SDV supports incremental routing updates. This allows a wireless network to dynamically adapt to routing changes. Moreover, unlike traditional distance vector protocols, SDV does not suffer from the count-to-infinity problem,[1] because the scope is typically small (*e.g.*, We evaluate a 1000-node network with 32 beacons, and its average scope is 3.35 and maximum scope is 13. This implies routing loops can be detected within 13 hops).

4.3.4 Inter-Cluster Routing: Resilient Beacon Distance Vector (RBDV)

To support routing across clusters, each node is required to know its distances to all beacons. This can be achieved by constructing a spanning tree rooted from each beacon nodes to every other node in the network. Flooding beacon packets reliably is important to the routing performance, because loss of beacon packets

[1]The count-to-infinity problem is that when a link fails, it may take a long time (on the order of network diameter) before the protocol detects the failure. During the interim routing loops may exist.

may introduce errors in estimating the closest beacon and its distance, and degrade the performance of S4. We develop a simple approach to enhance resilience of beacon packets.

Routing state construction and maintenance: To construct routing state for beacon nodes, every beacon periodically broadcasts beacon packets, which are flooded throughout the network. Every node then keeps track of the shortest hop count and next-hop towards each beacon.

Since beacon packets are broadcast and typical MAC protocols (*e.g.*, CC1000 used in sensor motes) do not provide reliability for broadcast packets, it is essential to enhance the resilience of beacon packets at the network layer. Our idea is to have a sender retransmit the broadcast packet P until T neighbors have forwarded P or until the maximum retry count $Retry_{max}$ is reached. T and $Retry_{max}$ provide a tradeoff between overhead and reliability. In our evaluation, we use $Retry_{max} = 3$, $T = 100\%$ for beacon nodes, and $T = 1/3$ for non-beacon nodes. $T = 100\%$ for a beacon node is used because all neighbors of the beacon nodes should forward the beacon packet. In comparison, for a non-beacon node c, only a subset of c's neighbors are farther away from the beacon than c and need to forward the beacon packet received from c. Therefore we use a smaller T for non-beacon nodes.

4.3.5 Distance Guided Local Failure Recovery (DLF)

Wireless networks are subject to bursty packet losses and frequent topology changes. To provide high routing success rate and low routing stretch even in the presence of frequent topology changes and node/link failures, we develop a simple

and effective local failure recovery based on distance vectors.

Overview: A node s retransmits a packet when it does not receive an ACK within a retransmission timeout. When R retransmissions fail, s broadcasts a *failure recovery request*, which contains (i) the next hop s used, (ii) whether destination d is included in s's local cluster, and (iii) the distance to d if s's cluster includes d, or the distance to d's beacon otherwise. Upon hearing the failure requests, s's neighbors attempt to recover the packet locally. Our goal is to select the neighbor that is the closest to the destination as s's new next-hop; meanwhile the selection process should be cheap and easily distributed.

S4 uses distance guided local failure recovery to prioritize neighbors' responses based on their scoped distance vectors. Each node uses its priority to determine the time it needs to wait before sending *failure recovery response*. We further exploit broadcast nature of wireless medium to avoid implosion of recovery responses.

Distance guided local failure recovery: Our goal is to prioritize neighbors based on their distances to the destination so that the nodes closest to the destination can take over the forwarding. The problem is non-trivial, because the distance to the destination is not always available. When the destination is outside the local cluster, a neighbor only knows the distance to the destination's closest beacon, but not the distance from that beacon to the destination.

To address the issues, each node computes its priority using the algorithm in Figure 4.2. It involves two main scenarios. In the first scenario, s's local cluster

```
// Priorities from highest to lowest: 1, 2, 3, 4
if(d ∈ C(s))
  if(d ∈ C(self)) // d is in s's and self's clusters
    priority = δ(self, d) − δ(s, d) + 2;
  else // d is only in s's cluster
    priority = 4;
  end
else if(d ∈ C(self)) // d is only in self's cluster
    priority = 1;
else // self is outside s's and d's clusters
  priority = δ(self, L(d)) − δ(s, L(d)) + 3;
end
```

Figure 4.2: Computing priority using scoped distance vectors and beacon distance vectors

contains the destination d. This information is available in s's failure recovery request. Then s's neighbor is assigned one of the four priorities using the following rules. The neighbors that have d in their clusters are assigned the top 3 priorities, since they can directly route towards destination using the shortest path. In this case, each neighbor knows its distance to the destination, and assigns itself a priority based on the difference between $\delta(self, d)$ and $\delta(s, d)$. Neighbors whose local clusters do not contain the destination are assigned the fourth priority, which is the lowest.

In the second case, when s's cluster does not contain the destination d, only the neighbors that have d in their clusters are assigned the highest priority, since they can directly route towards the destination. The other nodes are assigned priorities by comparing their distances to the beacon with $\delta(s, L(d))$.

A sender s selects the neighbor from which it receives the response first as the new next-hop. By assigning each neighbor i with a timer $priority(i) \times m + rand$, a higher priority node sends the response earlier and is thus favored as the new next-hop node. To avoid collisions, we add a small random timer $rand$ to the priority-based timer so that different nodes are likely to respond at different times even when assigned the same priority. To avoid response implosion, upon hearing a failure response to s from someone else, the current node cancels its own pending recovery response if any. Our evaluation uses $m = 50ms$, and $rand$ ranges from 0 to 49ms.

Node failures vs. link failures: The above scheme works well for link failures. When a node fails, all the links to and from the failed nodes are down. Therefore we need to avoid using nodes that use the failed nodes as next hop. This can be done by letting the sender specify the failed node. Only the nodes that use different next hop from the failed node will attempt to recover. In practice, it is difficult to distinguish between a link failure and a node failure. Always assuming a node failure may unnecessarily prune out good next-hops. So we first optimistically assume that the next hop does not fail, only the link is down. Therefore we allow nodes with the same next hop to recover the packet. When the number of failed attempts pass a threshold, we prevent the nodes from using the same next hop to recover the packet.

4.3.6 Other Design Issues

Location directory: So far we assume that the source knows which beacon node is closest to the destination. In practice, such information may not be directly available. In such situation, the source can apply the location directory scheme de-

scribed in BVR [26] to lookup such information. More specifically, beacon nodes are responsible for storing the mapping between non-beacon nodes and their closest beacons. The closest beacon information for node i is stored at $H(i)$, where H is a consistent hash function that maps $nodeid$ to $beaconid$. The source contacts the beacon node whose ID is $H(dest)$ to obtain the closest beacon to $dest$. The storage cost of location directory is much smaller in S4 than that in BVR, because the source in S4 only needs to know the closest beacon to its destination while the source in BVR needs to know the distance between its destination and all beacon nodes. Moreover, in S4 when destination d is in s's cluster, no location lookup is required since s knows the shortest path to d, whereas BVR as well as other geographic routing schemes always require location lookup on a new destination. Such property is especially beneficial when traffic exhibits locality (*i.e.*, nodes close to each other are more likely to communicate).

Beacon maintenance: When a beacon fails, S4 applies distance guided local failure recovery to temporarily route around the failure. If the failure persists, we can apply the beacon maintenance protocol proposed in [26] to select a new beacon. Beacon maintenance is not the focus of this chapter. Instead, we focus on the routing performance during the transient period after failures occur.

Link quality: Link quality significantly affects routing performance. We define link quality as the delivery rate of packet on the link in a given direction. In S4, each node continuously monitors its links to/from its neighbors. We adopt a passive link estimator layer developed in [26, 106] for estimating link quality. When a node receives a beacon packet or SDV update, it first checks if *both* the forward and

reverse link qualities of the sender are above a threshold (30% is used in our current implementation). Only those updates from a sender with good link quality in both directions will be accepted.

4.4 TOSSIM Evaluation

We have implemented a prototype of S4 in nesC language for TinyOS [41]. The implementation can be directly used both in TOSSIM simulator [63] and on real sensor motes. In this section, we evaluate the performance of S4 using extensive TOSSIM packet-level simulations. By taking into account of actual packet transmissions, collisions, and losses, TOSSIM simulation results are more realistic.

Our evaluation considers a wide range of scenarios by varying the number of beacon nodes, network sizes, network densities, link loss rates, and traffic demands. More specifically, we consider two types of network densities: a high density with an average node degree of 16.6 and a low density with an average node degree of 7.6. We use both lossless links and lossy links that are generated by *LossyBuilder* in TOSSIM. Note that even when links are lossless, packets are still subject to collision losses. In addition, we examine two types of traffic: a single flow and 5 concurrent flows. The request rate is one flow per second for single-flow traffic, and 5 flows per second for 5-flow traffic. The simulation lasts for 1000 seconds. So the total number of routing requests is 1000 for single-flow traffic, and 5000 for 5-flow traffic. We compare S4 with BVR, the implementation of which is available from the public CVS repository of TinyOS.

The performance metrics we are interested in are:

- Routing Success Rate
- Routing Stretch
- Control Traffic Overhead
- Routing State
- Node Load of Data Traffic

4.4.1 Routing Performance

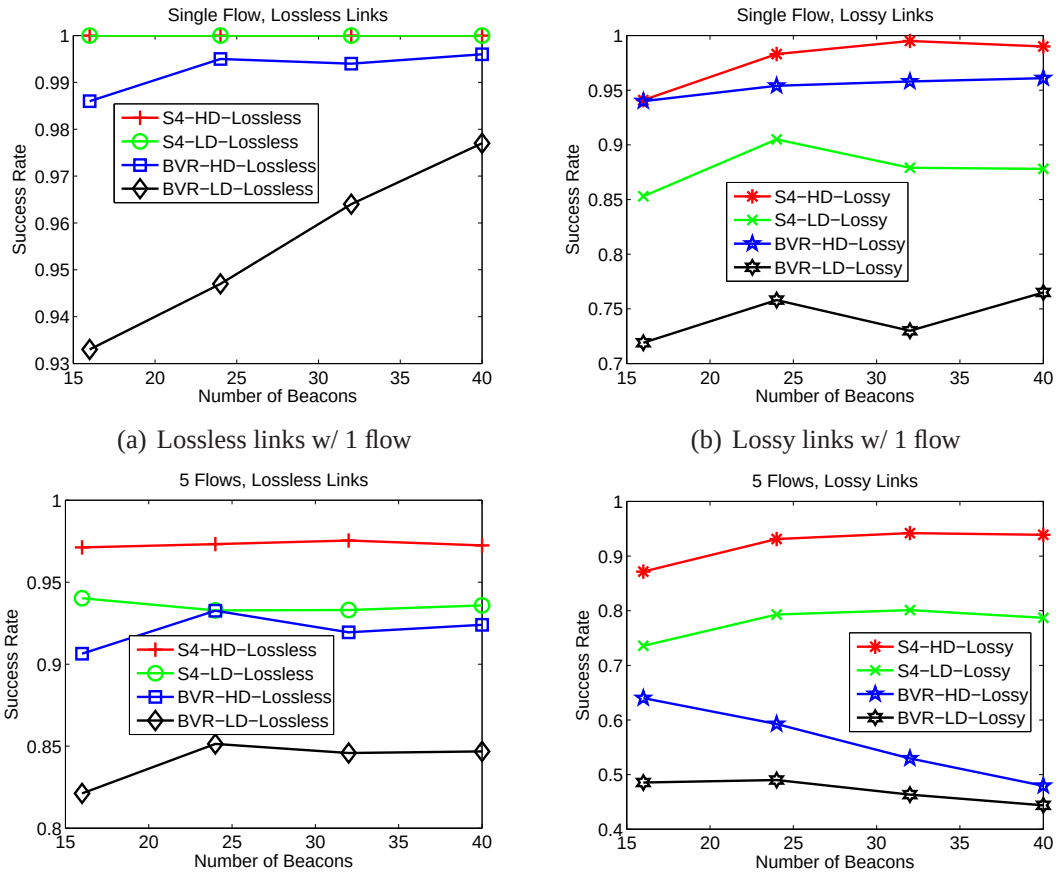
First we compare S4 with BVR under stable network conditions. To reach stable network conditions, we let each node periodically broadcast RBDV and SDV packets every 10 seconds. Data traffic is injected into the network only after route setup is finished. BVR uses scoped flooding after a packet falls back to the beacon closest to the destination and greedy forwarding still fails, whereas S4 uses the distance guided failure recovery scheme to recover failures. To make a fair comparison, in both BVR and S4 beacon nodes periodically broadcast and build spanning trees, and RBDV is turned off in S4.

4.4.1.1 Varying the Number of Beacons

We vary the number of beacon nodes from 16 to 40 while fixing the total number of nodes to 1000.

Routing success rate: We study 4 configurations: a single flow with lossless links, a single flow with lossy links, 5 flows with lossless links, and 5 flows with lossy

links. Figure 4.3 shows the results of all 4 configurations. "HD" and "LD" curves represent results under high and low network densities, respectively.



(a) Lossless links w/ 1 flow

(b) Lossy links w/ 1 flow

(c) Lossless links w/ 5 flows

(d) Lossy links w/ 5 flows

Figure 4.3: Compare routing success under different numbers of beacons, network densities and traffic patterns.

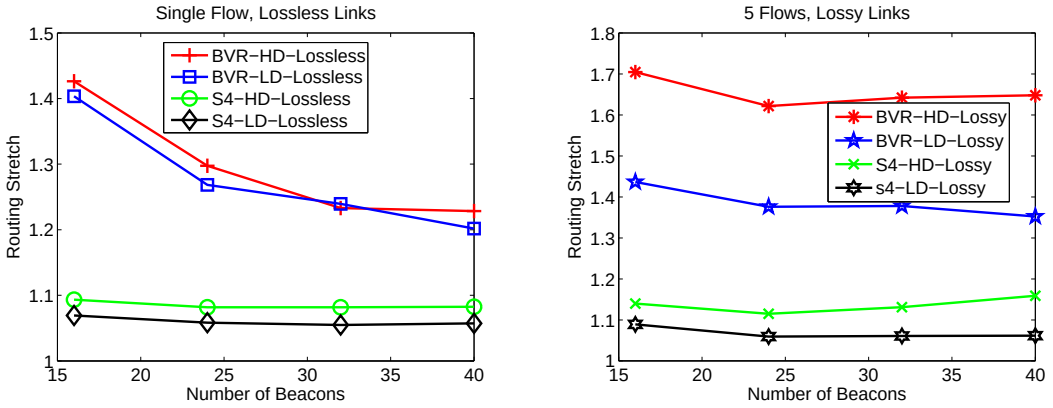
We make the following observations. First, under lossless links with 1 flow, S4 always achieves 100% success rate. In comparison, BVR achieves close to 100% success only in high-density networks, but its success rate reduces to 93% under low network density with 16 beacons. Why does BVR not provide delivery

guarantee even under perfect channel condition? The reason is that, scoped flooding is invoked after a packet is stuck at the fallback beacon, and scoped flooding could cause packet collisions and reduce packet delivery rate. Second, under lossy links with 5 flows, packet losses are common, and the performance of both S4 and BVR degrades. Nevertheless, S4 still achieves around 95% routing success rate in high-density networks, while success rate of BVR drops dramatically. The large drop in BVR is because its scoped flooding uses broadcast packets, which have no reliability support from MAC layer; in comparison, data packets are transmitted in unicast under S4, and benefit from link layer retransmissions. Third, the success rate is lowest under low-density networks, with lossy links and 5 flows. Even in this case S4 achieves 70% - 80% success rate, while the success rate of BVR is reduced to below 50%.

Routing stretch: Figure 4.4 compares the average routing stretch of S4 and BVR. The average routing stretch is computed only for the packets that have been successfully delivered. Although the worst stretch of S4 is 3, its average stretch is only around 1.1 - 1.2 in all cases. In comparison, BVR has significantly larger routing stretch: its average routing stretch is 1.2 - 1.4 for 1 flow, and 1.4 - 1.7 for 5 flows. Moreover its worst routing stretch (not shown) is 8.

Another interesting observation is that the routing stretch in S4 is consistently low regardless of the number of beacon nodes, whereas the routing stretch of BVR is more sensitive to the number of beacons. The reason is that S4 reason that S4 routing stretch remains low under various numbers of

does not have clear impact on routing stretch. This can be explained as



(a) Lossless links w/ 1 flow

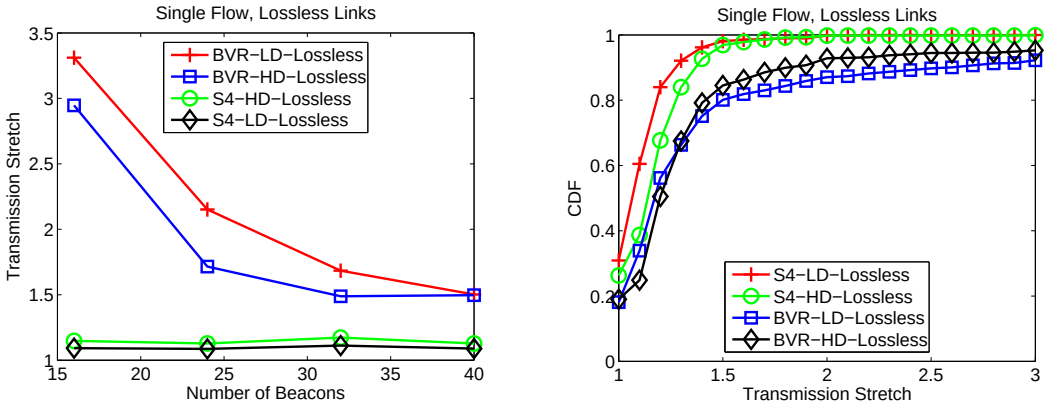
(b) Lossy links w/ 5 flows

Figure 4.4: Compare routing stretch under different numbers of beacons, network densities, and traffic patterns.

follows. On one hand, fewer beacon nodes result in larger local clusters. Intra-cluster routing, which is always optimal, happens more frequently. On the other hand, more beacon nodes provide more opportunities for better inter-cluster routing (in the extreme case where all nodes are beacon nodes, the routing stretch is always 1).

Transmission Stretch: As shown in Figure 4.5(a), the transmission stretch of S4 is close to its routing stretch, while the transmission stretch of BVR is much larger than its routing stretch due to its scoped flooding. Figure 4.5(b) shows CDF of transmission stretches under 32 beacon nodes. We observe that the worst-case transmission stretch in S4 is 3, and most of the packets have transmission stretch very close to 1.

Control traffic overhead: Compared with BVR, S4 introduces extra control traffic of SDV to construct routing tables for local clusters. To evaluate this overhead, we



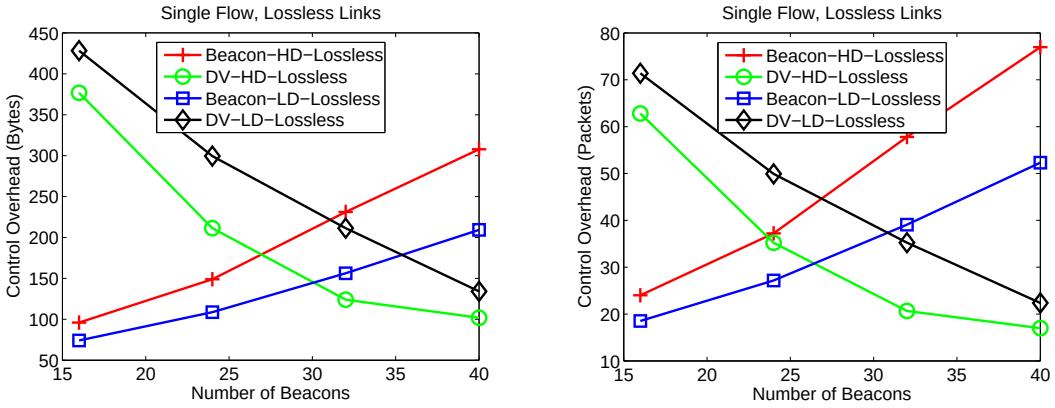
(a) Average transmission stretch (b) CDF of transmission stretch

Figure 4.5: Transmission stretch comparison

count the average control traffic (in bytes and number of packets) that each node generates under lossless links and a single flow. We separate the global beacon traffic and local SDV traffic. The results are shown in Figure 4.6. Note that beacon traffic overhead is the same for both S4 and BVR.

We can see that when the number of beacons is small, the SDV traffic dominates, since the cluster sizes are relatively large in such case. As the number of beacons increases, the amount of SDV traffic decreases significantly. In particular, when there are 32 beacons ($\approx \sqrt{1000}$), the amount of SDV traffic is comparable to the amount of global beacon traffic.

Routing state: We compare routing state of S4 and BVR as follows. For S4, the routing state consists of a beacon routing table and a local cluster table. For BVR, the routing state consists of a beacon routing table and a neighbor coordinate table. We first compare the total amount of routing state in bytes between S4 and BVR.



(a) Control traffic in Bytes

(b) Control traffic in packets

Figure 4.6: Control traffic overhead under different numbers of beacons and network densities

Figure 4.8(a) shows the average routing state over all nodes. We make the following observations. First, network density has little impact on the routing state of S4, but has large impact on BVR. This is because in S4 the local cluster sizes are not sensitive to network density (when density increases, the scope tends to decrease), while in BVR each node stores the coordinates of its neighbors and its routing state increases with density. Second, the amount of routing state in BVR increases with the number of beacons. In comparison, S4's routing state does not necessarily increase with the number of beacons, since increasing the number of beacons reduces the local cluster size. Third, when the number of beacons is 32 ($\approx \sqrt{1000}$) or above, the routing state in S4 is less than BVR. Similar results have been observed in other TOSSIM configurations.

Figure 4.8(b) further shows the number of entries in beacon routing table, local cluster table and neighbor coordinate table. The beacon table curves of S4

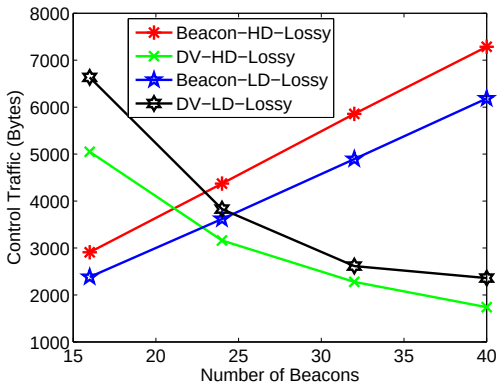


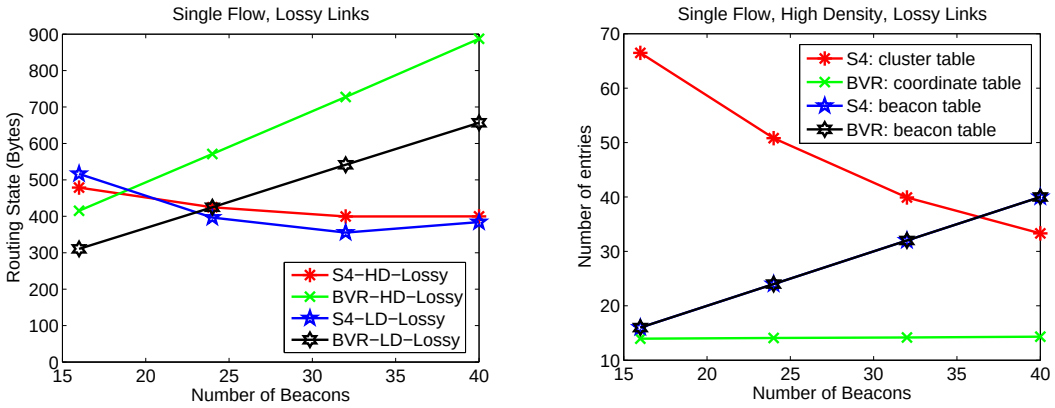
Figure 4.7: control traffic overhead w/ lossy links (5 flows)

and BVR overlap, since it is common for both. Note that although the coordinate tables in BVR have fewer entries than the cluster tables in S4, the total size of the coordinate tables are generally larger since the size of each coordinate table entry is proportional to the number of beacons.

Table 4.1 shows maximum routing state of S4 and BVR under high density and low density. The maximum number of routing entries is around 4.5 times of $\sqrt{1000}$ (the expected average cluster size), but still an order of magnitude smaller than 1000 (the flat routing table size) in shortest path routing. This suggests that random beacon selection does a reasonably good job in limiting worst-case storage cost.

	max S4 state (B)	max BVR state (B)	max S4 routing entries
HD	680	960	136
LD	715	920	143

Table 4.1: Maximum routing state of S4 and BVR



(a) Average routing state

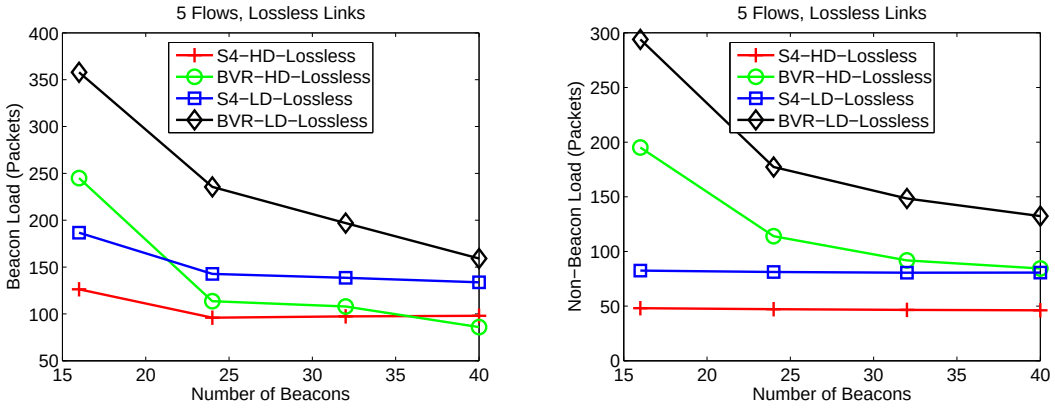
(b) number of routing table entries

Figure 4.8: Routing state comparison under different numbers of beacons and network densities with lossy links (single flow)

Node load: Figure 4.9 shows the average number of packets that each node transmits, under lossless links and 5-flow traffic. Figure 4.9(a) shows the beacon node load, and Figure 4.9(b) shows non-beacon node load. We observe that in S4 both beacon nodes and non-beacon nodes experience lower load than those nodes in BVR. This is due to lower routing stretch and transmission stretch in S4. In addition, we observe that in S4, the beacon load is within a factor of 1.5-2 of non-beacon load, which means the load is reasonably balanced among beacon and non-beacon nodes. Similar results are observed under single flow traffic.

4.4.1.2 Varying Network Size

We also evaluate the performance and scalability of S4 when the network size changes from 100 to 4000. For each network size N, we select $K \approx \sqrt{N}$ nodes as beacon nodes. We only include results under lossless links and a single



(a) Beacon load (b) Non-beacon load

Figure 4.9: Node load of data traffic under different numbers of beacons and network densities with lossless Links (5 flows)

flow. The results for other configurations are similar.

Figure 4.11(a) shows the average transmission stretch of S4 and BVR under different network sizes. The error bars represent 5- and 95- percentiles. S4 achieves smaller transmission stretches and smaller variations in the stretches. In BVR, packets experience higher medium stretch and higher stretch variation due to greedy forwarding and scoped flooding.

Figure 4.11(b) shows the average routing state. For both S4 and BVR, the routing state tends to increase with $O(\sqrt{N})$. This suggests both S4 and BVR are scalable with network sizes. In particular, even when the network size is 4000, majority of nodes can store the routing state in a small portion of a 4KB RAM (the RAM size on Mica2 motes we experimented with). Moreover, S4 uses less routing state than BVR when the number of beacon nodes is $\sqrt{N}$, because the coordinate table size in BVR is linear to the number of beacon nodes.

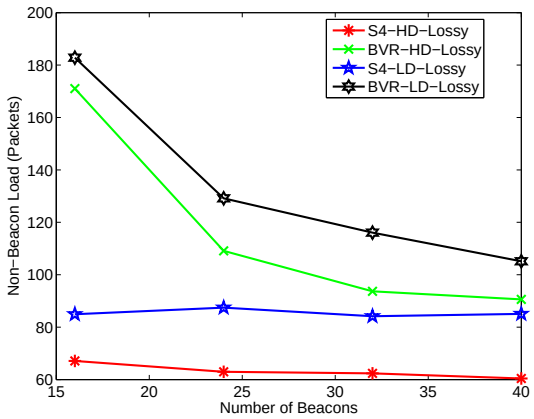
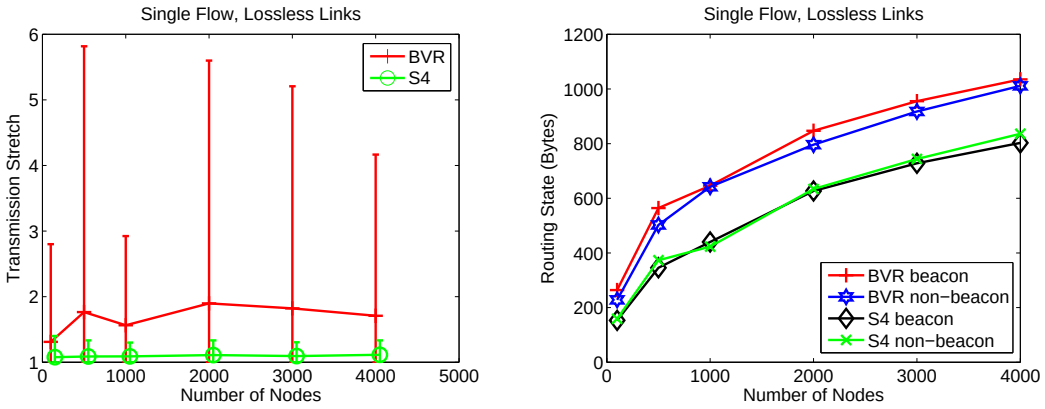


Figure 4.10: non-beacon load of data traffic w/ lossy links (5 flows)



(a) Transmission stretch (b) Routing state

Figure 4.11: Comparison under different network sizes

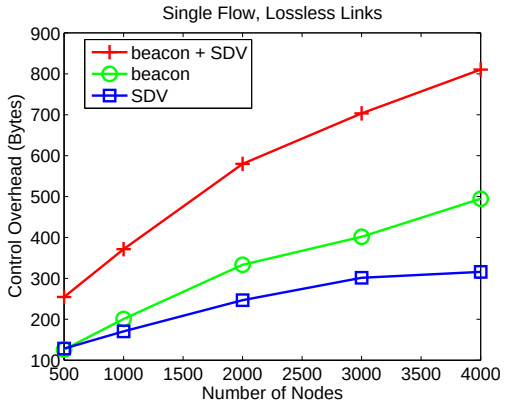


Figure 4.12: Average control traffic overhead comparison under different network sizes

Figure 4.12 shows the average control traffic generated over all nodes under lossless links and a single flow. The three curves represent total traffic, beacon traffic, and local SDV control traffic, respectively. The amount of local traffic is consistently smaller than that of beacon traffic under $\sqrt{N}$ beacon nodes. Since beacon traffic is the same in S4 and BVR, the total control traffic for route construction in S4 is comparable to that of BVR. The difference is further reduced when traffic for location directory setup is included.

	success rate	routing stretch	transmission stretch	control traffic (B)	routing state (B)
S4	1	1.07	1.08	96	158
BVR	0.994	1.20	1.31	46	232

Table 4.2: Performance comparison in 100-node networks.

To further study the performance of S4 in smaller networks, we compare S4 and BVR in networks of 100 nodes. We include the results for the case of

single flow traffic with lossless links. Results are similar for other configurations. Table 4.2 shows that in 100-node networks S4 outperforms BVR in terms of routing success rate, routing stretch, transmission stretch, and routing state. S4 incurs more control overhead than BVR due to the extra SDV traffic, though its overall control traffic (after including location directory setup traffic) is still comparable to that of BVR.

4.4.2 Impact of RBDV

Next we evaluate resilient beacon distance vector (RBDV). We turn off periodic transmissions of beacon and SDV messages so that the failed transmissions of these messages have to be recovered using RBDV but not using periodic beacon transmissions. This is an interesting scenario to consider because we want to minimize the frequency of periodic broadcasts while still achieving high delivery rate. Each beacon broadcasts once. Other nodes who receive a beacon packet further broadcast it. Similarly, a non-beacon node broadcasts its own scoped distance vector once. A node further broadcasts a SDV only if it is inside the scope.

We simulate for single-flow data traffic with lossless links, and compare the routing success rate between the case with and without RBDV. In both cases, DLF is enabled. Packet collisions are common when nodes broadcast beacon packets or scoped distance vectors. As shown in Figure 4.13, without RBDV, the success rate is around 90%. With RBDV, the success rate is improved to close to 100% because RBDV helps to improve accuracy of the routing tables.

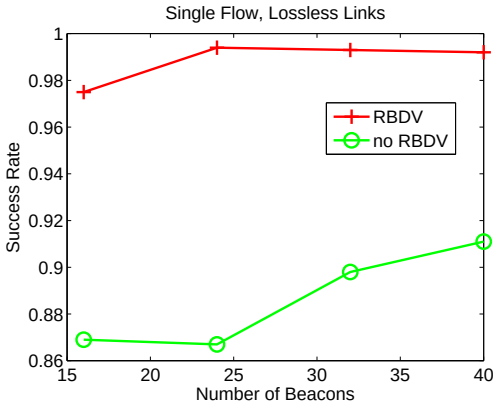


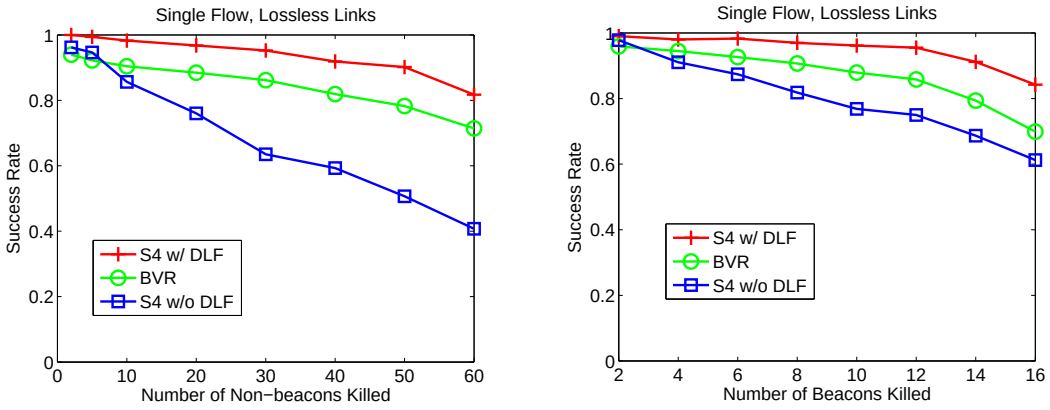
Figure 4.13: Impact of RBDV on success rate (1000 nodes, low density)

4.4.3 Impact of Node Failures

To evaluate the performance of S4 under node failures, we randomly kill a certain number of nodes right after the control traffic is finished. Different from the experiments in [26], we start node failures from the beginning, *i.e.* the control traffic is also subject to node failures. We distinguish between beacon and non-beacon failures,

Figure 4.14 shows that failure recovery can significantly increase the success rate under both non-beacon and beacon failures. DLF in S4 is more effective than the scoped flooding in BVR for the following reasons. First, scoped flooding results in packet collisions. Second, S4 uses unicast for data transmissions and benefits from link layer retransmissions. Third, if some node between the beacon and destination fails, DLF can recover such failures, while scoped flooding cannot.

Next we compute the average routing stretch over all successfully delivered packets. As we expect, packets going through failure recovery take longer than

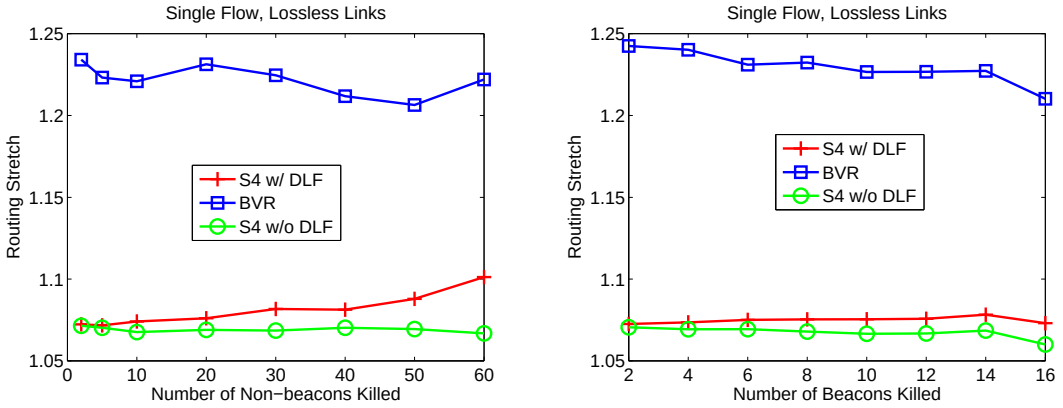


(a) Random non-beacon failures (b) Random beacon failures

Figure 4.14: Impact of DLF on success rate (1000 nodes, 32 beacons, low density)

normal paths. Interestingly, as shown in Figure 4.15, the average routing stretch is only slightly higher than the case of no failure recovery, which indicates the robustness of S4.

Summary Our TOSSIM evaluation further confirms that S4 is scalable to large networks: the average routing state scales with $O(\sqrt{N})$ in an N-node network. The average routing and transmission stretches in S4 are around 1.1-1.2. This is true not only in lossless networks under single flow traffic, but also under lossy wireless medium, packet collisions arising from multiple flows, and significant failures. This demonstrates that S4 is efficient and resilient. In comparison, the performance of BVR is sensitive to wireless channel condition. Even under loss-free networks, it may not provide 100% delivery guarantee due to possible packet collisions incurred in scoped flooding. Its routing and transmission stretches also increase with wireless losses and failures.



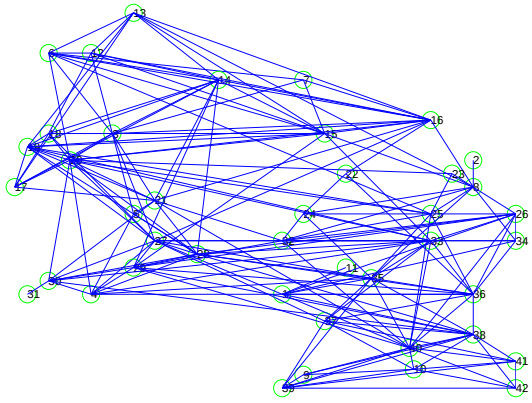
(a) Random non-beacon failures (b) Random beacon failures

Figure 4.15: Impact of DLF on routing stretch (1000 nodes, 32 beacons, low density)

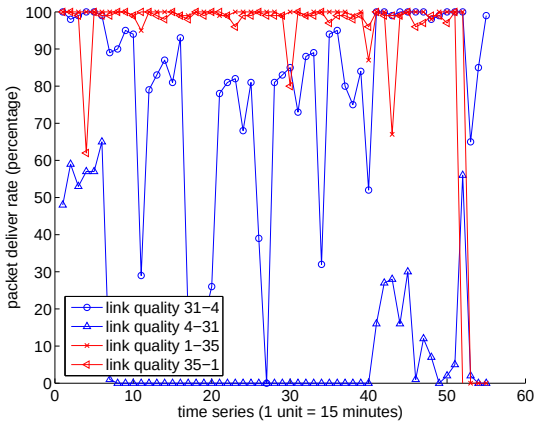
4.5 Testbed Evaluation

We have deployed the S4 prototype on a testbed of 42 $mica2$ motes with 915MHz radios on the fifth floor of ACES building at UT Austin. While the testbed is only moderate size and cannot stress test the scalability of S4, it does allow us to evaluate S4 under realistic radio characteristics and failures. We adjust the transmission power to -17dBm for all control and data traffic to obtain an interesting multi-hop topology. With such a power level, the testbed has a network diameter of around 4 to 6 hops, depending on the wireless link quality. 11 motes are connected to the MIB600 Ethernet boards that we use for logging information. They also serve as gateway nodes to forward commands and responses for the remaining 31 battery-powered motes. [2]

[2]Unfortunately, we are unable to compare S4 against BVR in our testbed. Current BVR implementation requires all motes have Ethernet boards connected to send and receive routing commands. However our testbed only has 11 motes with Ethernet connections, which would make the evaluation



(a) topology snapshot



(b) Link quality

Figure 4.16: Testbed measurement

Figure 4.16(a) shows a snapshot of the network topology. We measure packet delivery rates by sending broadcast packets on each mote one by one. Two motes have a link if the delivery rates on both directions are above 30%. Because no two nodes will broadcast packets at the same time, the measurement result is optimistic in the sense that channel contention and network congestion is not considered. The average node degree is 8.7. We observe that a short geographic distance between two motes does not necessarily lead to good link quality. Some of the links are very asymmetric and their qualities vary dramatically over time. As shown in Figure 4.16(b), some of the links are highly asymmetric and their qualities vary dramatically over time. For example, the link qualities between motes 4 and 31 fluctuate as time goes by and are quite asymmetric, while link qualities between motes 1 and 15 are fairly stable to 100% delivery rate, until in the last one hour when they suddenly drop to almost 0%. Such link characteristics allow us to stress test the performance and resilience of S4.

4.5.1 Routing Performance

We randomly preselect 6 nodes out of 42 nodes as beacon nodes for S4. The distance from any node to its closest beacon is at most 2 hops. After 10 minutes of booting up all the motes, we randomly select source and destination pairs to evaluate routing performance. The sources are selected from all 42 motes and the destinations are selected from the 11 motes that are connected to the Ethernet boards. All destinations dump the packet delivery confirmation through UART to

less interesting.

time period	# pkts per sec	routing success rate
0 - 70.1 min	1	99.9%
70.1 - 130.2 min	2	99.1%

Table 4.3: Routing success rate in the 42-node testbed.

the PC for further analysis. For each routing request, unless the source is connected to an Ethernet board, we choose the gateway mote that is the closest to the source to forward a command packet. The command packet is sent with the maximum power level, and up to 5 retransmissions so that the source is very likely to receive it. Upon receiving the routing request, the source will send back a response packet with the maximum power level and potential retransmissions, to acknowledge successful reception of the routing request. Each routing request is tagged with a unique sequence number to make the operation idempotent. The data packet will be sent (with the reduced power level) after the command traffic to avoid interference.

We send routing requests at 1 packet per second for the first 70 minutes (altogether 4210 packets), and then double the sending rate thereafter for another 60 minutes (altogether 7701 packets). As shown in Table 4.3, the routing success rate is 99.1-99.9%, and consistent over time. This demonstrates the resilience of S4 in a real testbed.

Next we use multiple constant bit rate (CBR) flows to increase the network load. In each multiple flow test, we randomly pick n source destination pairs, and instrument the sources to send consecutive packets at the rate of 1 packet per s seconds. This is essentially having n/s random flows per second. The flows start after a predefined idle period to avoid potential collisions with the command traffic.

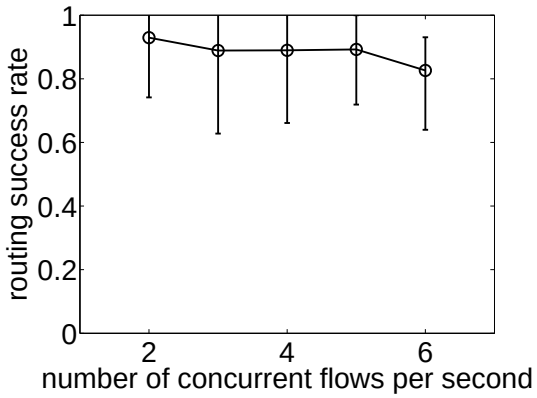


Figure 4.17: Routing success rate under multiple concurrent flows

We choose $s = 2$, and test up to 6 concurrent flows (*i.e.*, n is up to 12). For each experiment, we repeat it for 10 times. Figure 4.17 plots the median routing success rates in different flow settings. The error bars indicate the best-case and worst-case routing success rate. We see the median success rate gracefully degrades with an increasing number of concurrent flows. Our log collected from the gateway motes indicates that some of the failures are due to the limitation of single forwarding buffer per node. Such failure happens when two or more flows try to concurrently route through the same node. Note that this is not a protocol limitation in S4. We could remove many such failures by having a more complete implementation with multiple forwarding buffers, which will be part of our future work.

Finally we study the routing efficiency of S4. Note that it is impossible to calculate the true routing stretch in a real wireless network because the topology is always changing and the packet loss rates depend on the traffic pattern so that the optimal routes are changing, too. Instead, we compare S4 against the *pseudo optimal hop count* metric. The pseudo optimal hop count of a route is defined as

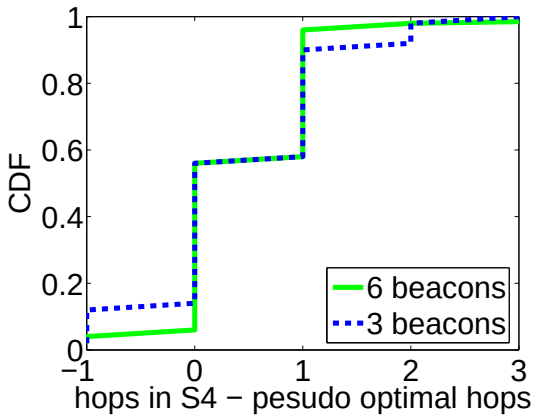


Figure 4.18: CDF of the hop count difference to pseudo optimal

the shortest path length in a *snapshot* of the network topology. In our experiment, we use broadcast-based active measurement to obtain the pairwise packet delivery rates before the routing test starts. The delivery rates are averaged over 1-hour measurement period. Note that the real optimal routes could be either better or worse than the pseudo optimal ones due to topology changes, and the delivery rates tend to be optimistic due to no packet collision in the measurement. The routing tests follow the measurement within 30 minutes. We randomly select source and destination pairs and send routing requests at 1 packet per second for 5000 seconds. Then we change the number of beacons from 6 to 3, and repeat the same test. The shortest paths from the topology snapshot are computed offline. Figure 4.18 shows that more than 95% of the routes are within 1-hop difference from the pseudo optimal hops under 6 beacons. Interestingly, S4 sometimes achieves better performance than the pseudo optimal scheme. This is because during the 5000-second routing experiment, S4 adapts to the change of topology so that it can take advantages of

new links and reduce path lengths. The number of beacons also has both positive and negative effects on routing performance. When fewer beacons are selected, the nodes tend to have larger routing tables so that more nodes can be reached via the shortest paths; however, having fewer beacons also leads to more control traffic so that the link estimator will have a more pessimistic estimation on link quality due to packet collision. Underestimating link quality apparently hurts the routing performance.

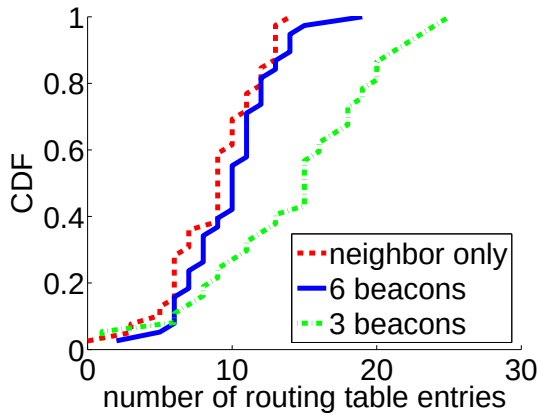


Figure 4.19: Routing table size

In the same experiment, we also study the routing state per node in S4. Figure 4.19 compares the numbers of local routing table entries used under 6 and 3 beacons. Using 6 beacons yields smaller routing tables. A node in S4 has local routing state towards its neighbor unless the neighbor is a beacon node. Therefore the number of routing entries at each node is generally larger than the number of its neighbors. We find that on average, when 6 beacons are used, the routing table has only 3 more entries than a typical neighborhood table, which suggests that the

routing state in S4 is small.

4.5.2 Routing Under Node Failures

To stress test the resilience of S4, we artificially introduce node failure in our testbed. We randomly select non-gateway motes to kill one by one, and study the routing performance. We send one routing request per second for 50 minutes, altogether generating 3000 packets. The source node is randomly selected from the current live nodes and the destination is one of the gateway motes. Note that we do not start any SDV update or beacon broadcast after the initial setup stage in order to study the effectiveness of the failure recovery mechanism alone.

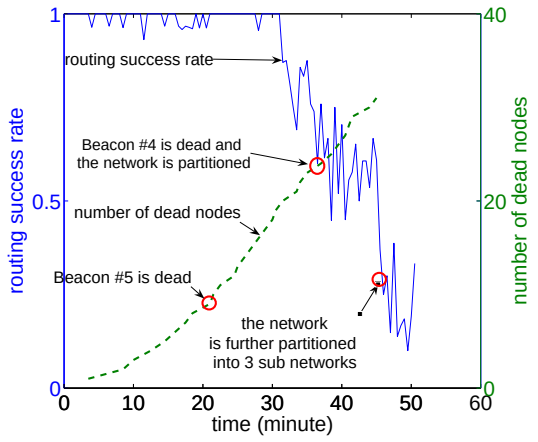


Figure 4.20: Routing performance under node failure

As shown in Figure 4.20, in the first 30 minutes, even when 20 motes are killed, including a beacon node, the routing success rate is still close to 100%. The routing success rate starts to drop after 30 minutes, due to congestion at some bottleneck links. When the second beacon is killed, the network is partitioned and

more routing failures are expected. The third major performance degradation occurs after all 31 non-gateway motes are dead, which causes further network partitions. These results show that S4 is resilient to failures.

Summary Our evaluation in the 42 node testbed shows that S4 achieves close to 100% routing success rate in a normal condition with a single flow. Meanwhile S4 degrades gracefully with an increasing number of packet collisions (in multiple concurrent flows) and node failures.

Chapter 5

Conclusions and Future Work

In this section, we summarize the contributions of the dissertation and give directions for future work.

5.1 Conclusions

MWNs bring new fundamental challenges to network management. First, multihop connections make localization more difficult since nodes are not in direct range of anchor points. Second, interactions among nodes add significant complexity in modeling and understanding wireless interference. Third, effective routing control is hard when network scale increases. To address these challenges, we develop (i) probabilistic region-based localization algorithms, (ii) a general model of wireless interference, and (iii) a scalable routing protocol for large MWNs.

- **Probabilistic Region-Based Localization:** We propose distributed, probabilistic region-based algorithms for localization under multihop connections. The algorithms take simple binary connectivity measurements as basic location constraints. The mutual constraints among nodes are modeled in the computations of probability distributions. To improve accuracy, the algorithms can also take additional measurements, such as finer-grained connec-

tivity, layout maps and angle information. Furthermore, we extend the basic algorithms to enhance robustness of localization. Through extensive simulations, we verify that our algorithms can achieve high accuracy with low computation cost, and tolerate significant measurement errors.

The accuracy of our algorithms is attributed to their probabilistic nature. It handles uncertainty and errors better than deterministic approaches that estimate locations as single points. With iterative computations of probability distributions, nodes can take advantage of estimations from each other for refinement even they are multiple hops away.

- **A General Model of Wireless Interference:** To study wireless interference and its impact, we develop a general interference model. It takes simple RSSI measurements from real networks and models the interdependencies among transmissions and receptions of the nodes. It allows us to accurately estimate the throughput and goodput in static multi-hop wireless networks. Compared to existing measurement-based models, our model can handle arbitrary number of senders, unicast transmissions, and non-saturated traffic demands. It provides a powerful tool to conduct what-if analysis and helps find optimal network configurations, such as power and channel assignment.

  At the core of our model is a p-persistent CSMA approximation to 802.11 DCF. This methodology can be generalized and applied to model MAC protocols other than 802.11. The main difference is that individual node's state transition probabilities should be computed based on the MAC protocol to be modeled. The N-node Markov chain framework would still be applicable.

- **Small State and Small Stretch Routing:** For routing control in large scale MWNs, we present a new routing protocol, Small State and Small Stretch (S4), which jointly minimizes routing state and routing stretch. S4 is a unique addition to the routing protocol design space. Specifically, it is the first routing protocol that achieves a worst-case constant routing stretch of 3, using $O(\sqrt{N})$ routing state per node, in an N-node large scale wireless networks. And it employs a distance guided local failure recovery scheme to significantly enhance network resilience to failures. We evaluate S4 with both simulations and testbed experiments, and demonstrate that S4 simultaneously achieves scalability, efficiency, and reliability.

  S4 adapts the idea of compact routing. It shows an example of protocol design guided by existing results in theory research area. How to make theoretical results work in reality is not trivial. Many new problems arise in real implementation. In our case, we combine new techniques with basic compact routing to obtain a practical routing protocol for large scale MWNs.

5.2 Future Work

Management in MWNs has received little attention until recently. This dissertation makes preliminary efforts in addressing some of the challenges involved in this area. There are many interesting directions to explore for future work. We briefly describe several possible topics for measurement, modeling, and control.

- **Measurement:** Measurement has been more and more important in networking research, since it reflects the true state of underlying networks. Automatic, efficient, and accurate measurements are desired in MWNs management. For our interference model, the RF profiling still needs improvement. Currently, we estimate RSS using average RSSI. This estimation may be biased for lossy links, because we can only directly measure RSSI for received packets. How to estimate RSS for lossy links is an interesting subject for future work.

- **Modeling:** Our interference model estimates throughput and goodput of links. In practice, traffic may traverse more than one link. What users see, and hence care more about, is end-to-end performance of a data flow. A future direction to extend the current model is to estimate end-to-end throughput and goodput based on the estimations of links. For the extended model, we are given end-to-end traffic demands and routing, from which we need to derive per hop demands. Then we can apply the current model to estimate per hop performance, and finally estimate the end-to-end performance.

- **Control:** In current implementation of S4, we use the simplest routing metric, hop count. However, hop count metric is not always correlated with network performance. In fact, since long hops usually have poor quality, a route of fewer but longer hops may have higher loss rate. It is therefore preferred to use more performance-relevant metrics. To guarantee the worst-case stretch of 3, the routing metrics must be additive and symmetric. ETX satisfies the

conditions and is a promising choice of metrics for S4. However, ETX of a link may change due to variations of channel conditions. It is a challenging issue to maintain routing states up-to-date with ETX.

Mobility poses similar problem to routing control. Node movement can cause frequent topology changes, which may invalidate existing routing states. Mobility is common in MANETs. Mesh networks may also have low to medium node mobility. Therefore, it is important for routing to support mobility. With current design of S4, it may require frequent broadcast of beacon distance vector messages and scoped distance vector messages, which invoke too much overhead, to maintain routing states consistent with topology changes. New mechanisms are needed to reduce such overhead while still achieving the trade-off among scalability, efficiency, and reliability.

Finally, there are many other controls to explore in future, such as channel assignment, power control, node placement, etc. Each of them has significant impact on network performance. The output from the interference model can guide these controls to achieve optimal performance.

Bibliography

[1] U. G. Acer, S. Kalyanaraman, and A. A. Abouzeid. Weak state routing for large scale dynamic networks. In *Proc. of MobiCom'07*, Sept. 2007.

[2] S. Agarwal, J. Padhye, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Measurement and estimation of link interference in static multi-hop wireless networks. In *Proc. of Internet Measurement Conference*, Oct. 2005.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.

[4] Ian F. Akyildiz, Xudong Wang, and Weilin Wang. Wireless mesh networks: a survey. *Computer Networks and ISDN Systems*, 47(4):445–487, March 2005.

[5] ALERT systems. http://www.alertsystems.org/.

[6] P. Bahl and V. N. Padmanabhan. RADAR: an in-building rf-based user location and tracking system. In *Proc. of IEEE Infocom'00*, Mar. 2000.

[7] E. M. Belding-Royer. Multi-level hierarchies for scalable ad hoc routing. *Wireless Networks*, Sept. 2003.

[8] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. In *IEEE Journal on Selected Areas in Communications*, March 2000.

[9] P. Biswas, T. Lian, T. Wang, and Y. Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Trans. on Sensor Networks*, May 2006.

[10] Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia. Routing with guaranteed delivery in ad-hoc wireless networks. In *Proc. of Dial-M'99*, Aug. 1999.

[11] Raffaele Bruno, Marco Conti, and Enrico Gregori. Mesh networks: commodity multihop ad hoc networks. *IEEE Communications Magazine*, 43(3):123–131, March 2005.

[12] M. Caesar, M. Castro, E. B. Nightingale, G. O'Shea, and A. Rowstron. Virtual ring routing: Network routing inspired by DHTs. In *Proc. of ACM SIGCOMM*, Sept. 2006.

[13] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: application driver for wireless communications technology. In *Proc. of ACM SIGCOMM Workshop on Data Communications*, Aug. 2001.

[14] Hoon Chang, Vishal Misra, and Dan Rubenstein. A general model and analysis of physical layer capture in 802.11 networks. In *Proc. of IEEE INFOCOM*, Apr. 2006.

[15] Y. Cheng, J. Bellardo, P. Benko, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: Solving the puzzle of enterprise 802.11 analysis. In *SIGCOMM*, 2006.

[16] Douglas E. Comer. *Automated Network Management Systems*. Prentice Hall, 2007.

[17] J. A. Costa, N. Patwari, and A. O. Hero III. Distributed weighted-multidimensional scaling for node localization in sensor networks. *ACM Trans. on Sensor Networks*, Feb. 2006.

[18] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. ACM MOBICOM*, 2003.

[19] L. Cowen. Compact routing with minimum stretch. *J. of Algorithms*, 2001.

[20] S. R. Das, C. E. Perkins, and E. M. Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proc. of IEEE INFOCOM*, Mar. 2000.

[21] DistMesh – a simple mesh generator in MATLAB. http://www-math.mit.edu/ persson/mesh/.

[22] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proc. of ACM MOBICOM*, Sept. - Oct. 2004.

[23] E. Elnahrawy, X. Li, and R. P. Martin. The limits of localization using signal strength: A comparative study. In *Proc. of the IEEE Conference on Sensor and Ad Hoc Communication Networks (SECON)*, Oct. 2004.

[24] T. Eren, D. Goldenberg, W. Whitley, Y. R. Yang, A. S. Morse, B. D.O. Anderson, and P. N. Belhumeur. Rigidity, computation, and randomization of network localization. In *Proc. of IEEE INFOCOM*, Apr. 2004.

[25] L. F. Fenton. The sum of lognormal probability distributions in scatter transmission systems. *IRE Trans. Commun. Syst.*, CS-8, 1960.

[26] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, David Culler, Scott Shenker, and Ion Stoica. Beacon Vector Routing: Scalable Point-to-Point Routing in Wireless Sensornets. In *Proc. of NSDI'05*, May 2005.

[27] Hannes Frey and Ivan Stojmenovic. On delivery guarantees of face and combined greedy-face routing in ad hoc and sensor networks. In *Proc. of MOBICOM 2006*, Sept. 2006.

[28] K. Gabriel and R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, pages 259–278, 1969.

[29] A. Galstyan, B. Krishnamachari, K. Lerman, and S. Pattem. Distributed online localization in sensor networks using a moving target. In *Proc. of the International Symposium on Information Processing in Sensor Networks*, Apr. 2004.

[30] Y. Gao, J. Lui, and D. M. Chiu. Determining the end-to-end throughput capacity in multi-hop networks: Methodolgy and applications. In *Proc. of ACM SIGMETRICS*, Jun. 2006.

[31] M. Garetto, J. Shi, and E. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *Proc. of ACM MOBICOM*, Aug. - Sept. 2005.

[32] M. Gerla, X. Hong, and G. Pei. Landmark routing for large ad hoc wireless networks. In *Proc. of Globecom*, Nov. 2000.

[33] D. K. Goldenberg, A. Krishnamurthy, W. C. Maness, Y. R. Yang, A. Young, A. S. Morse, A. Savvides, and B. D.O. Anderson. Network localization in partially localizable networks. In *Proc. of IEEE Infocom*, Mar. 2005.

[34] Global positioning system standard positioning service specification. *United States Coast Guard Navigation Center*, June 1995.

[35] S. Guha, R. N. Murty, and E. G. Sirer. Sextant: A unified framework for node and event localization in sensor networks. In *Proc. of ACM MOBIHOC*, May 2005.

[36] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and mitigating the impact of rf interference on 802.11 networks. In *Proc. of Sigcomm'07*, Aug. 2007.

[37] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2), Mar. 2000.

[38] Zygmunt J. Haas, Marc R. Pearlman, and Prince Samar. The zone routing protocol (ZRP) for ad hoc networks. Internet-draft, IETF MANET Working Group, July 2002.

[39] A. Haeberien, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Oractical robust localization over large-scale 802.11 wireless networks. In *Proc. of MobiCom'04*, Sept. 2004.

[40] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *IEEE Computer*, Aug 2001.

[41] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *Proc. of the ASPLOS*, 2000.

[42] L. Hu and D. Evans. Localization for mobile sensor networks. In *Proc. of ACM MOBICOM*, Sept. 2004.

[43] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proc. of ACM MOBICOM*, Sept. 2003.

[44] D. Johnson, D. Maltz, and J. Broch. DSR: The dynamic source routing protocol for multihop wireless ad hoc networks. In *Ad Hoc Networking*, 2001.

[45] B. Karp and H.T. Kung. Greedy perimeter stateless routing for wireless networks. In *Proc. of ACM MOBICOM*, Aug. 2000.

[46] A. Kashyap, S. Ganguly, and S. R. Das. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *Proc. of MobiCom'07*, Sept. 2007.

[47] V. Kawadia and P. R. Kumar. Principles and protocols for power control in ad hoc networks. In *IEEE Journal on Selected Areas in Communications (JSAC)*, January 2005.

[48] Young-Jin Kim, Ramesh Govindan, Brad Karp, and Scott Shenker. Geographic routing made practical. In *Proc. of NSDI'05*, May 2005.

[49] Young-Jin Kim, Ramesh Govindan, Brad Karp, and Scott Shenker. On the pitfalls of geographic face routing. In *Proc. of the 3rd ACM/SIGMOBILE Intl. Workshop on Foundation of Mobile Computing (DIAL-M-POMC)*, 2005.

[50] L. Kleinrock and J. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. In *NTC '78; National Telecommunications Conference*, Dec. 1978.

[51] A. Kochut, A. Vasan, A. U. Shankar, and A. Agrawala. Sniffing out the correct physical layer capture model in 802.11b. In *Proc. of ICNP*, Oct. 2004.

[52] M. Korkea-aho. Context-aware applications survey. *Internetworking Seminar*, Apr. 2000.

[53] David Kotz, Calvin Newport, and Chip Elliott. The mistaken axioms of wireless-network research. Technical Report TR2003-467, Dartmouth College, Computer Science, Jul. 2003.

[54] Jacek B. Krawczyk and Steffan Berridge. Relaxation algorithms in finding Nash equilibria. In Computational Economics from Economics Working Paper Archive at WUSTL, Jul. 1997.

[55] U. Kubach and K. Rothermel. Exploiting location information for infostation-based hoarding. In *Proc. of ACM MOBICOM*, Jul. 2001.

[56] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: Of theory and practice. In *Proc. of ACM PODC*, 2003.

[57] F. Kuhn, R. Wattenhofer, and A. Zollinger. Asymptotically optimal geometric mobile ad-hoc routing. In *Proc. of Dial-M'02*, 2002.

[58] A. Kumar, E. Altman, D. Miorandi, and M. Goyal. New insights from a fixed-point analysis of single cell IEEE 802.11 WLANs. *IEEE/ACM Trans. on Networking*, Jun. 2007.

[59] V. S. A. Kumar and M. Marathe. Algorithmic aspects of capacity in wireless networks. In *Proc. of ACM SIGMETRICS*, Jun. 2005.

[60] L. Lazos and R. Poovendran. SeRLoc: Robust localization for wireless sensor networks. *ACM Trans. on Sensor Networks*, Aug. 2005.

[61] B. Leong, S. Mitr, and B. Liskov. Path vector face routing: Geographic routing with local face information. In *Proc. of IEEE ICNP*, Nov. 2005.

[62] Ben Leong, Barbara Liskov, and Robert Morris. Geographic routing without planarization. In *Proc. of NSDI'06*, May 2006.

[63] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. TOSSIM: accurate and scalable simulation of entire tinyos applications. In *Proc. of ACM SenSys*, 2003.

[64] J. Li, C. Blake, D. S. J. De Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proc. of MOBICOM*, 2001.

[65] M. Li and Y. Liu. Rendered path: Range-free localization in anisotropic sensor networks with holes, Sept. 2007.

[66] Z. Li, S. Nandi, and A. K. Gupta. Improving fairness in IEEE 802.11 using enhanced carrier sensing. In *IEEE Communications*, Oct. 2004.

[67] H. Lim, L. Kung, J. C. Hou, and H. Luo. Zero-configuration, robust indoor localization: Theory and experimentation. In *Proc. of Infocom'06*, Apr. 2006.

[68] D. Madigan, E. Elnahrawy, and R. P. Martin. Bayesian indoor positioning systems. In *Proc. of IEEE Infocom'05*, Mar. 2005.

[69] David Malone, Ken Duffy, and Doug Leith. Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions. *IEEE/ACM Transactions on Networking*, 15(1), February 2007.

[70] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, and W. Arbaugh. A client-driven approach for channel management in wireless LANs. In *IEEE Infocom*, 2006.

[71] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *Proc. of ACM SENSYS*, Nov. 2004.

[72] J. Newsome and D. Song. GEM: Graph embedding for routing and data-centric storage in sensor networks without geographic information. In *Proc. of ACM SenSys'03*, Nov. 2003.

[73] D. Niculescu and B. Nath. VOR base stations for indoor 802.11 positioning. In *Proc. of MobiCom'04*, Sept. 2004.

[74] L. M. S. C. of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Standard 802.11*, 1999.

[75] C. C. Paige and M. A. Saunders. LSAR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Soft.*, 1982.

[76] G. Pei, M. Gerla, X. Hong, and C. Chiang. A wireless hierarchical routing protocol with group mobility. In *Proc. of WCNC'99*, Sept. 1999.

[77] R. Peng and M. L. Sichitiu. Angle of arrival localization for wireless sensor networks. In *Proc. of Secon'06*, Sept. 2006.

[78] R. Peng and M. L. Sichitiu. Probabilistic localization for outdoor wireless sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, Jan. 2007.

[79] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proc. of ACM SIGCOMM*, 1994.

[80] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, Feb. 1999.

[81] P. O. Persson and G. Strang. A simple mesh generator in MATLAB. *SIAM Review*, Jun. 2004.

[82] A. Post and D. Johnson. Self-organizing hierarchical routing for scalable ad hoc networking. *Rice CS Technical Report TR04-433*, 2004.

[83] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proc. of ACM MOBICOM*, Aug 2000.

[84] The Qualnet simulator. http://www.scalable-networks.com/.

[85] V. Ramadurai and M. L. Sichitiu. Localization in wireless sensor networks: A probabilistic approach. In *Proc. of ICWN'03*, Jun. 2003.

[86] A. Rao, Sylvia Ratnasamy, Christos Papadimitriou, Scott Shenker, and Ion Stoica. Geographic routing without location information. In *Proc. of ACM MOBICOM*, Sept. 2003.

[87] S. Ray, D. Starobinski, A. Trachtenberg, and R. Ungrangsi. Robust location detection with sensor networks. In *IEEE JSAC (Special Issue on Fundamen-*

tal Performance Limits of Wireless Sensor Networks), Vol. 22, No. 6, pp. 1016-1025, Aug. 2004.

[88] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference. In *Proc. of ACM SIGCOMM*, Sept. 2006.

[89] MIT Roofnet. http://www.pdos.lcs.mit.edu/roofnet/.

[90] E. Rozner, Y. Mehta, A. Akella, and L. Qiu. Traffic-aware channel assignment in enterprise wireless networks. In *Proc. of ICNP*, Oct. 2007.

[91] M. Rudafshani and S. Datta. Localization in wireless sensor networks. In *Proc. of IPSN'07*, Apr. 2007.

[92] A. Savvides, C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proc. of ACM MOBICOM*, Jul. 2001.

[93] A. Savvides, H. Park, and M. B. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proc. of WSNA'02*, Sep. 2002.

[94] S. Schwartz and Y. Yeh. On the distribution function and moments of power sums with lognormal distributions. *Bell Systems Technical Journal*, 61, 1982.

[95] Y. Shang and W. Ruml. Improved MDS-based localization. In *Proc. of IEEE INFOCOM*, Apr. 2004.

[96] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz. Localization from mere connectivity. In *Proc. of ACM MOBIHOC*, Jun. 2003.

[97] M. L. Sichitiu, V. Ramadurai, and P. Peddabachagari. Simple algorithm for outdoor localization of wireless sensor networks with inaccurate range measurements. In *Proc. of ICWN'03*, Jun. 2003.

[98] D. C. Steere, A. Baptista, D. McNamee, C. Pu, and J. Walpole. Research challenges in environmental observation and forecasting systems. In *Proc. of 6th International Conference on Mobile Computing and Networking*, 2000.

[99] M. Takai, J. Martin, and R. Bagrodia. Effects of wireless physical layer modeling in mobile ad hoc networks. In *Proc. of ACM MOBIHOC*, Oct. 2001.

[100] M. Thorup and U. Zwick. Approximate distance oracles. In *Proc. of ACM STOC*, 2001.

[101] M. Thorup and U. Zwick. Compact routing schemes. In *Proc. of SPAA'01*, Jul. 2001.

[102] G. Toussaint. The relative neighborhood graph of a finite planar set. *Pattern Recognition*, pages 261–268, 1980.

[103] P. F. Tsuchiya. The landmark hierarchy: a new hierarchy for routing in very large networks. In *Proc. of Sigcomm'88*, Aug. 1988.

[104] Nitin H. Vaidya. Tutorial on mobile ad hoc networks: routing, MAC, and transport issue. In *Mobicom*, 2001.

[105] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Trans. on Information Systems*, Jan. 1992.

[106] Alec Woo, Terence Tong, and David Culler. Taming the underlying challenges of reliable multhop routing in sensor networks. In *Proc. of ACM SenSys'03*, Nov. 2003.

[107] J. Zhang, T. Yan, J. A. Stankovic, and S. H. Son. Thunder: Towards practical, zero cost acoustic localization for outdoor wireless sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, Jan. 2007.

Vita

Feng Wang received his B.S. and M.E. degrees in Computer Science from University of Science and Technology of China in 1997 and 2000, respectively. He received the M.S. degree in Computer Sciences from University of Texas at Austin in 2002. His research interests are in wireless networks, particularly network management and protocol design in multihop wireless networks. He held summer internship at Microsoft Research Asia in Beijing, in 2002. He has been a Teaching Assistant since 2000 and a Graduate Research Assistant since 2005 in Computer Sciences department in University of Texas at Austin.

Permanent address: 1801 S. Lakeshore Blvd., Apt. 219
Austin, Texas 78741

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.